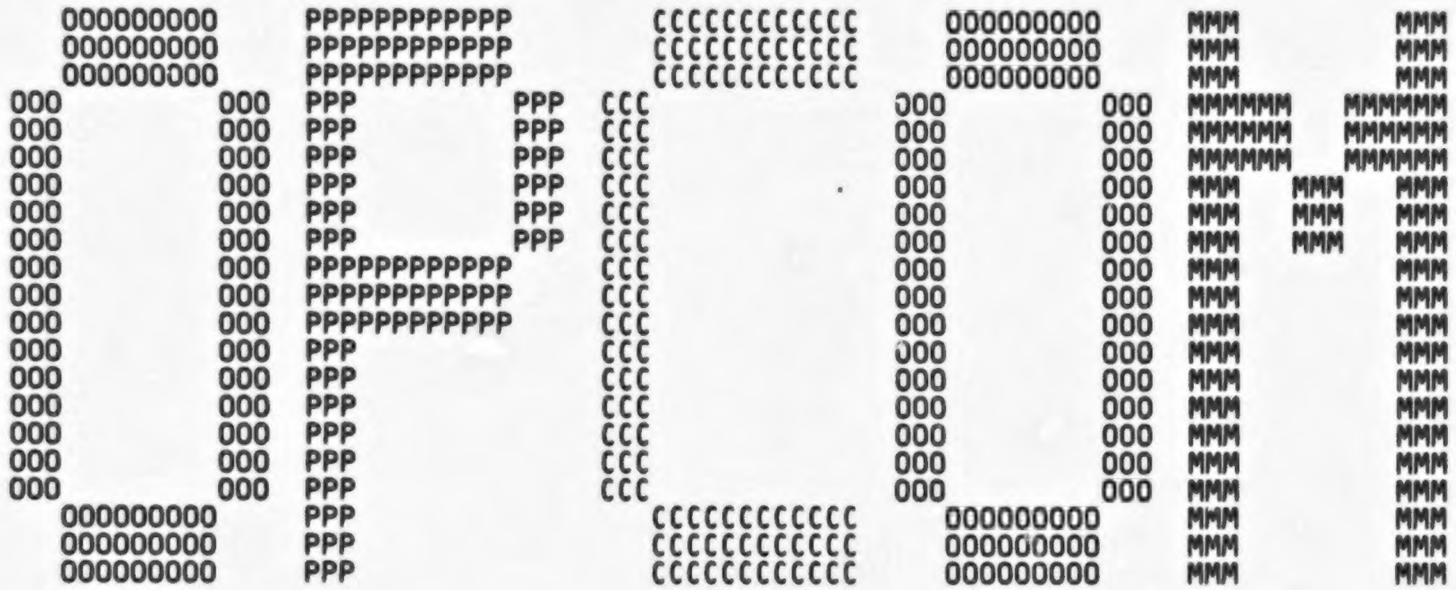

_$2

Sym

ALL

ASC

BOD
BOD
BOD
BOD
BOD
BOD
BOD
BOD
BUG
BYP
CAN
CAN
CAN
CHE
CHE

CLU
CLU
CLU
CLU
CLU
CLU
CLU

CLU
CLU
CLU
CLU
CLU
CLU
CLU

CLU
CLU

```
  000000    PPPPPPPP    EEEEEEEEEE  RRRRRRRR    UU      UU  TTTTTTTTTT    IIIIII    LL
  000000    PPPPPPPP    EEEEEEEEEE  RRRRRRRR    UU      UU  TTTTTTTTTT    IIIIII    LL
00      00  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
00      00  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
00      00  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
00      00  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
00      00  PPPPPPPP    EEEEEEEE    RRRRRRRR    UU      UU      TT          II      LL
00      00  PPPPPPPP    EEEEEEEE    RRRRRRRR    UU      UU      TT          II      LL
00      00  PP          EE          RR  RR      UU      UU      TT          II      LL
00      00  PP          EE          RR  RR      UU      UU      TT          II      LL
00      00  PP          EE          RR    RR    UU      UU      TT          II      LL          ....
00      00  PP          EE          RR    RR    UU      UU      TT          II      LL          ....
  000000    PP          EEEEEEEEEE  RR      RR  UUUUUUUUUU      TT      IIIIII    LLLLLLLLLL  ....
  000000    PP          EEEEEEEEEE  RR      RR  UUUUUUUUUU      TT      IIIIII    LLLLLLLLLL  ....

LL          IIIIII      SSSSSSSS
LL          IIIIII      SSSSSSSS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II        SSSSSS
LL            II        SSSSSS
LL            II              SS
LL            II              SS
LL            II              SS
LL            II              SS
LLLLLLLLLL  IIIIII    SSSSSSSS
LLLLLLLLLL  IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE  OPC$OPERUTIL    (
;     2          0002  0                          LANGUAGE (BLISS32),
;     3          0003  0                          IDENT = 'V04-000'
;     4          0004  0                         ) =
;     5          0005  0
;     6          0006  0 !*****************************************************************************
;     7          0007  0 !*                                                                           *
;     8          0008  0 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;     9          0009  0 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
;    10          0010  0 !*  ALL RIGHTS RESERVED.                                                     *
;    11          0011  0 !*                                                                           *
;    12          0012  0 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    13          0013  0 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  0 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    15          0015  0 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    16          0016  0 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    17          0017  0 !*  TRANSFERRED.                                                             *
;    18          0018  0 !*                                                                           *
;    19          0019  0 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    20          0020  0 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    21          0021  0 !*  CORPORATION.                                                             *
;    22          0022  0 !*                                                                           *
;    23          0023  0 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    24          0024  0 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    25          0025  0 !*                                                                           *
;    26          0026  0 !*                                                                           *
;    27          0027  0 !*****************************************************************************
;    28          0028  0
;    29          0029  0 !++
;    30          0030  0 ! FACILITY:
;    31          0031  0 !
;    32          0032  0 !       OPCOM
;    33          0033  0 !
;    34          0034  0 ! ABSTRACT:
;    35          0035  0 !
;    36          0036  0 !       This module contains the general utility routines used
;    37          0037  0 !       to manipulate operator control blocks.  These routines
;    38          0038  0 !       are used freely throughout OPCOM's request handlers.
;    39          0039  0 !
;    40          0040  0 ! Environment:
;    41          0041  0 !
;    42          0042  0 !       VAX/VMS operating system.
;    43          0043  0 !
;    44          0044  0 ! Author:
;    45          0045  0 !
;    46          0046  0 !       Steven T. Jeffreys
;    47          0047  0 !
;    48          0048  0 ! Creation date:
;    49          0049  0 !
;    50          0050  0 !       March 10, 1981
;    51          0051  0 !
;    52          0052  0 ! Revision history:
;    53          0053  0 !
;    54          0054  0 !       V03-004 CWH3169         CW Hobbs                        5-May-1984
;    55          0055  0 !               Second pass for cluster-wide OPCOM:
;    56          0056  0 !                - Use queued brkthru mechanism to send messages.
;    57          0057  0 !                - Add DVI$_ code to SHARE_FULL_DEVNAME calls.
```

```
;    58          0058  0 !
;    59          0059  0 !       V03-003 CWH3003         CW Hobbs                        16-Sep-1983
;    60          0060  0 !               Add a flag that a disable is in progress to prevent recursive
;    61          0061  0 !               disables.  Change $BRDCST to $BRKTHRU.  Comment out incomplete
;    62          0062  0 !               code for mailboxes as operators.
;    63          0063  0 !
;    64          0064  0 !       V03-002 CWH3002         CW Hobbs                        30-Jul-1983
;    65          0065  0 !               Various and sundry things to make OPCOM distributed
;    66          0066  0 !               across the cluster.
;    67          0067  0 !
;    68          0068  0 ! V03-001       STJ3034         Steven T. Jeffreys,     06-Oct-1982
;    69          0069  0 !               Check for dial-in terminal.  Treat it as a remote terminal.
;    70          0070  0 !
;    71          0071  0 ! V02-002       STJ0165         Steven T. Jeffreys,     08-Feb-1982
;    72          0072  0 !               Make references to library routines use general addressing mode.
;    73          0073  0 !
;    74          0074  0 !--
;    75          0075  0
;    76          0076  1 BEGIN                                           ! Start of OPERUTIL
;    77          0077  1
;    78          0078  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    79          0079  1 LIBRARY 'LIB$:OPCOMLIB';
;    80          0080  1
;    81          0081  1 FORWARD ROUTINE
;    82          0082  1         CHECK_OPER_COVERAGE : NOVALUE,          ! Check operator coverage on requests
;    83          0083  1         FIND_OPERATOR,                          ! Find a given operator RQCB
;    84          0084  1         IMPLICIT_DISABLE,                       ! Check for implict disable of an operator
;    85          0085  1         NOTIFY_LISTED_OPERATORS,                ! Notify the operators on the given list
;    86          0086  1         NOTIFY_OPERATOR,                        ! Send a message to an operator
;    87          0087  1         OPERUTIL_CLM_IMP_DISABLE : NOVALUE,     ! Implicit disable from another node
;    88          0088  1         UPD_OPER_CONTEXT,                       ! Update operator context
;    89          0089  1         VALID_OPERATOR;                         ! Check for valid operator device
;    90          0090  1
;    91          0091  1 BUILTIN
;    92          0092  1         INSQUE,                                 ! Insert entry onto a queue
;    93          0093  1         REMQUE;                                 ! Remove entry from a queue
;    94          0094  1
;    95          0095  1 EXTERNAL
;    96          0096  1         GLOBAL_STATUS : BITVECTOR [32],
;    97          0097  1         LCL_NOD : $ref_bblock,
;    98          0098  1         LCL_CSID;
;    99          0099  1
;   100          0100  1 EXTERNAL ROUTINE
;   101          0101  1         CLUSUTIL_SYSTEMID_EQUAL : JSB_R0R1,
;   102          0102  1         DUMP_LOG_FILE,
;   103          0103  1         REPLYBRD_BRKTHRU_QUEUE : NOVALUE;       ! Queue a $brkthru I/O
```

```
;  105     0104  1 GLOBAL ROUTINE CHECK_OPER_COVERAGE (OCD) : NOVALUE =
;  106     0105  1
;  107     0106  1 !++
;  108     0107  1 ! Functional descripton:
;  109     0108  1 !
;  110     0109  1 !       This routine will check all outstanding requests queued to
;  111     0110  1 !       a given OCD for proper operator coverage.  Any request that
;  112     0111  1 !       no longer has operator coverage will be canceled.  The requestor
;  113     0112  1 !       will receive a NOPERATOR cancelation message.  No operators are
;  114     0113  1 !       notified, since none are interested in the request.  The cancelation
;  115     0114  1 !       is, however, logged.
;  116     0115  1 !
;  117     0116  1 ! Input:
;  118     0117  1 !
;  119     0118  1 !       OCD     : Address of an OCD
;  120     0119  1 !
;  121     0120  1 ! Implicit Input:
;  122     0121  1 !
;  123     0122  1 !       None.
;  124     0123  1 !
;  125     0124  1 ! Output:
;  126     0125  1 !
;  127     0126  1 !       None.
;  128     0127  1 !
;  129     0128  1 ! Implicit Output:
;  130     0129  1 !
;  131     0130  1 !       None.
;  132     0131  1 !
;  133     0132  1 ! Side Effects:
;  134     0133  1 !
;  135     0134  1 !       None.
;  136     0135  1 !
;  137     0136  1 ! Routine Value
;  138     0137  1 !
;  139     0138  1 !       None.
;  140     0139  1 !--
;  141     0140  1
;  142     0141  2 BEGIN                                   ! Start of CHECK_OPER_COVERAGE
;  143     0142  2
;  144     0143  2 MAP
;  145     0144  2         OCD             : $ref_bblock;  ! OCD data structure
;  146     0145  2
;  147     0146  2 EXTERNAL ROUTINE
;  148     0147  2         DEALLOCATE_RQCB : NOVALUE,      ! Dispose of an RQCB
;  149     0148  2         FORMAT_MESSAGE,                 ! Format a message
;  150     0149  2         LOG_MESSAGE,                    ! Log an event
;  151     0150  2         SEND_REPLY;                     ! Send a reply to reply mailbox
;  152     0151  2
;  153     0152  2 LOCAL
;  154     0153  2         MESSAGE_VECTOR  : VECTOR [2,LONG],  ! Message info
;  155     0154  2         MCB             : $ref_bblock,      ! MCB data structure
;  156     0155  2         RQST_COUNT      : LONG,             ! Count of outstanding requests
;  157     0156  2         RQST            : $ref_bblock,      ! Pointer to current request RQCB
;  158     0157  2         NEXT_RQST       : LONG;             ! Pointer to next request RQCB
;  159     0158  2 !
;  160     0159  2 ! Set up the message info vector.
;  161     0160  2 !
```

```
;  162          0161  2 MESSAGE_VECTOR [0] = OPC$_NOPERATOR;                  ! Set message code
;  163          0162  2 MESSAGE_VECTOR [1] = 0;                               ! Set # of FAO arguements
;  164          0163  2 !
;  165          0164  2 ! Set up for the search loop.
;  166          0165  2 !
;  167          0166  2 NEXT_RQST = .OCD [OCD_L_RQSTFLINK];                   ! Get address of next RQCB
;  168          0167  2 RQST_COUNT = .OCD [OCD_W_RQSTCOUNT];                  ! Get count of requests
;  169          0168  2 WHILE (.RQST_COUNT GTR 0) DO
;  170          0169  3     BEGIN
;  171          0170  3     !
;  172          0171  3     ! Compare the request attention mask against the operator
;  173          0172  3     ! intererst mask for this OCD.  If there are no common
;  174          0173  3     ! bits, then the request does not have any operator coverage
;  175          0174  3     ! and must be canceled.
;  176          0175  3     !
;  177          0176  3     RQST = .NEXT_RQST;                               ! Get address of request RQCB
;  178          0177  3     NEXT_RQST = .RQST [RQCB_L_FLINK];                ! Get address of next RQCB
;  179          0178  4     IF  ((.RQST [RQCB_L_ATTNMASK1] AND .OCD [OCD_L_ATTNMASK1]) EQL 0)
;  180          0179  4     AND ((.RQST [RQCB_L_ATTNMASK2] AND .OCD [OCD_L_ATTNMASK2]) EQL 0)
;  181          0180  3     THEN
;  182          0181  4         BEGIN
;  183          0182  4         !
;  184          0183  4         ! Cancel the request.  This entails removing it from the OCD
;  185          0184  4         ! request queue, sending the cancel notice to the requestor,
;  186          0185  4         ! and deallocating the request RQCB.
;  187          0186  4         !
;  188          0187  4         REMQUE (.RQST, RQST);                        ! Dequeue the request
;  189          0188  4         OCD [OCD_W_RQSTCOUNT] = .OCD [OCD_W_RQSTCOUNT] - 1;
;  190          0189  4         FORMAT_MESSAGE (.RQST, MESSAGE_VECTOR);
;  191          0190  4         SEND_REPLY (.RQST);
;  192          0191  4         LOG_MESSAGE (.RQST);
;  193          0192  4         DEALLOCATE_RQCB (.RQST);
;  194          0193  3         END;
;  195          0194  3     RQST_COUNT = .RQST_COUNT - 1;
;  196          0195  2     END;
;  197          0196  2
;  198          0197  1 END;                                                 ! End of CHECK_OPER_COVERAGE


                                                        .TITLE  OPC$OPERUTIL
                                                        .IDENT  \V04-000\

                                                        .EXTRN  GLOBAL_STATUS, LCL_NOD
                                                        .EXTRN  LCL_CSID, CLUSUTIL_SYSTEMID_EQUAL
                                                        .EXTRN  DUMP_LOG_FILE, REPLYBRD_BRKTHRU_QUEUE
                                                        .EXTRN  DEALLOCATE_RQCB
                                                        .EXTRN  FORMAT_MESSAGE, LOG_MESSAGE
                                                        .EXTRN  SEND_REPLY

                                                        .PSECT  $CODE$,NOWRT,2

                                001C 00000              .ENTRY  CHECK_OPER_COVERAGE, Save R2,R3,R4       ; 0104
                             5E 04   C2 00002           SUBL2   #4, SP                                   ;
                 00058061    8F   DD 00005              PUSHL   #360545                                  ; 0161
                       04    AE   D4 0000B              CLRL    MESSAGE_VECTOR+4                         ; 0162
                     50 04   AC   D0 0000E              MOVL    OCD, R0                                  ; 0166
                     53 3C   A0   D0 00012              MOVL    60(R0), NEXT_RQST                        ;
```

```
                        54        3A   A0  3C 00016          MOVZWL  58(R0), RQST_COUNT                ; 0167
                                       44  15 0001A 1$:      BLEQ    3$                               ; 0168
                        52             53  D0 0001C          MOVL    NEXT_RQST, RQST                  ; 0176
                        53             62  D0 0001F          MOVL    (RQST), NEXT_RQST                ; 0177
                        50        04   AC  D0 00022          MOVL    0CD, R0                          ; 0178
                48      A0        5C   A2  D3 00026          BITL    92(RQST), 72(R0)                 ;
                                       2F  12 0002B          BNEQ    2$                               ;
                4C      A0        60   A2  D3 0002D          BITL    96(RQST), 76(R0)                 ; 0179
                                       28  12 00032          BNEQ    2$                               ;
                        52             62  0F 00034          REMQUE  (RQST), RQST                     ; 0187
                        50        04   AC  D0 00037          MOVL    0CD, R0                          ; 0188
                                  3A   A0  B7 0003B          DECW    58(R0)                           ;
                                4004   8F  BB 0003E          PUSHR   #^M<R2,SP>                       ; 0189
              0000G     CF             02  FB 00042          CALLS   #2, FORMAT_MESSAGE               ;
                                       52  DD 00047          PUSHL   RQST                             ; 0190
              0000G     CF             01  FB 00049          CALLS   #1, SEND_REPLY                   ;
                                       52  DD 0004E          PUSHL   RQST                             ; 0191
              0000G     CF             01  FB 00050          CALLS   #1, LOG_MESSAGE                  ;
                                       52  DD 00055          PUSHL   RQST                             ; 0192
              0000G     CF             01  FB 00057          CALLS   #1, DEALLOCATE_RQCB              ;
                                       54  D7 0005C 2$:      DECL    RQST_COUNT                       ; 0194
                                       BA  11 0005E          BRB     1$                               ; 0168
                                           04 00060 3$:      RET                                      ; 0197

; Routine Size:  97 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  200     0198  1 GLOBAL ROUTINE FIND_OPERATOR (RQCB, BLOCK) =
;  201     0199  1
;  202     0200  1 !++
;  203     0201  1 ! Functional description:
;  204     0202  1 !
;  205     0203  1 !       This routine will scan through the list(s) of operators
;  206     0204  1 !       known by OPCOM, and return the address of the operator
;  207     0205  1 !       RQCB if it is found.
;  208     0206  1 !
;  209     0207  1 ! Input:
;  210     0208  1 !
;  211     0209  1 !       RQCB    : Address of an RQCB that describes the operator
;  212     0210  1 !                 device that is being sought.
;  213     0211  1 !
;  214     0212  1 ! Implicit Input:
;  215     0213  1 !
;  216     0214  1 !       None.
;  217     0215  1 !
;  218     0216  1 ! Output:
;  219     0217  1 !
;  220     0218  1 !       BLOCK   : Contains the address of a longword to receive
;  221     0219  1 !                 the address of the known operator RQCB.
;  222     0220  1 !
;  223     0221  1 ! Implict output:
;  224     0222  1 !
;  225     0223  1 !       None.
;  226     0224  1 !
;  227     0225  1 ! Side effects:
;  228     0226  1 !
;  229     0227  1 !       If the operator is found, then the RQCB is provided
;  230     0228  1 !       with a pointer to the OCD.
;  231     0229  1 !
;  232     0230  1 ! Routine value:
;  233     0231  1 !
;  234     0232  1 !       TRUE    : If the operator is known to OPCOM
;  235     0233  1 !       FALSE   : If the operator is not known to OPCOM
;  236     0234  1 !--
;  237     0235  1
;  238     0236  2 BEGIN                                           ! Start of FIND_OPERATOR
;  239     0237  2
;  240     0238  2 MAP
;  241     0239  2         RQCB            : $ref_bblock;          ! RQCB data structure
;  242     0240  2
;  243     0241  2 EXTERNAL ROUTINE
;  244     0242  2         IMPLICIT_DISABLE;                       ! Check for implicit disable
;  245     0243  2
;  246     0244  2 EXTERNAL LITERAL
;  247     0245  2         MIN_SCOPE,                              ! Minimum scope value
;  248     0246  2         MAX_SCOPE;                              ! Maximum scope value
;  249     0247  2
;  250     0248  2 EXTERNAL
;  251     0249  2         OCD_VECTOR      : VECTOR;               ! Pointer to OCD structure
;  252     0250  2
;  253     0251  2 LOCAL
;  254     0252  2         OCD             : $ref_bblock,          ! OCD data structure
;  255     0253  2         OPER_RQCB       : $ref_bblock,          ! Operator RQCB structure
;  256     0254  2         OCD_INDEX       : LONG;                 ! Index into OCD_VECTOR
```

```
;  257     0255  2          OCD_COUNT       : LONG,                 ! Count of OCDs in the OCD list
;  258     0256  2          OPER_COUNT      : LONG,                 ! Count of operators in OCD list
;  259     0257  2          FOUND           : LONG;                 ! Boolean loop control
;  260     0258  2
;  261     0259  2      !
;  262     0260  2      ! Scan through the list of all known operators,
;  263     0261  2      ! looking for a match on the device name.
;  264     0262  2      ! The scan is started on the lowest privileged
;  265     0263  2      ! operator class and proceeds to the highest.
;  266     0264  2      !
;  267     0265  2      .BLOCK = 0;                                   ! Zero the output parameter
;  268     0266  2      FOUND = FALSE;                                ! Assume not found
;  269     0267  2      OCD_INDEX = MAX_SCOPE;                        ! Set higest (lowest privileged) scope value
;  270     0268  2      WHILE (.OCD_INDEX GEQ MIN_SCOPE) AND (NOT .FOUND) DO
;  271     0269  3          BEGIN
;  272     0270  3          !
;  273     0271  3          ! Scan the OCD list for each class of operator.
;  274     0272  3          !
;  275     0273  3          OCD = .OCD_VECTOR [(.OCD_INDEX - 1)*2];       ! Get OCD address
;  276     0274  3          OCD_COUNT = .OCD_VECTOR [(.OCD_INDEX - 1)*2+1];   ! Get count of known operators of this scope
;  277     0275  3          WHILE (NOT .FOUND) AND (.OCD_COUNT GTR 0) DO
;  278     0276  4              BEGIN
;  279     0277  4              !
;  280     0278  4              ! Scan the operator list for each OCD.
;  281     0279  4              !
;  282     0280  4              OPER_COUNT = .OCD [OCD_W_OPERCOUNT];      ! Get the count of operators in the list
;  283     0281  4              OPER_RQCB  = .OCD [OCD_L_OPERFLINK];      ! Get pointer to first operator in the list
;  284     0282  4              WHILE (.OPER_COUNT GTR 0) AND (NOT .FOUND) DO
;  285     0283  5                  BEGIN
;  286     0284  5                  !
;  287     0285  5                  ! Examine the device name for each operator in the list.
;  288     0286  5                  ! Compare the operator device names for equality.
;  289     0287  5                  ! Both device names are assumed to be in the DDCU format.
;  290     0288  5                  !
;  291     0289  5                  IF CH$EQL   (.OPER_RQCB [RQCB_L_OPER_LEN],
;  292     0290  5                               .OPER_RQCB [RQCB_L_OPER_PTR],
;  293     0291  5                               .RQCB [RQCB_L_OPER_LEN],
;  294     0292  5                               .RQCB [RQCB_L_OPER_PTR],
;  295     0293  5                               0
;  296     0294  5                              )
;  297     0295  5                  THEN
;  298     0296  6                      BEGIN
;  299     0297  6                      FOUND = TRUE;                 ! The operator is known to OPCOM
;  300     0298  6                      .BLOCK = .OPER_RQCB;          ! Save the RQCB address
;  301     0299  6                      RQCB [RQCB_L_OCD] = .OCD;     ! Save the OCD address
;  302     0300  6                      END
;  303     0301  5                  ELSE
;  304     0302  6                      BEGIN
;  305     0303  6                      OPER_RQCB = .OPER_RQCB [RQCB_L_FLINK];    ! Get link to next operator RQCB
;  306     0304  6                      OPER_COUNT = .OPER_COUNT - 1;             ! Decrement operator count
;  307     0305  5                      END;
;  308     0306  4                  END;
;  309     0307  4              OCD = .OCD [OCD_L_FLINK];                 ! Get address of next OCD
;  310     0308  4              OCD_COUNT = .OCD_COUNT - 1;               ! Decrement OCD count
;  311     0309  3              END;
;  312     0310  3      OCD_INDEX = .OCD_INDEX - 1;                       ! Decrement OCD_INDEX
;  313     0311  2      END;
```

```
;  314          0312  2
;  315          0313  2 !
;  316          0314  2 ! If the operator was found, make sure
;  317          0315  2 ! it has not been implicitly disabled.
;  318          0316  2 !
;  319          0317  2 IF .FOUND
;  320          0318  2 THEN
;  321          0319  2     FOUND = NOT (IMPLICIT_DISABLE (.OPER_RQCB));
;  322          0320  2 RETURN (.FOUND);                          ! Return status of search
;  323          0321  2
;  324          0322  1 END;                                      ! End of FIND_OPERATOR


                                                        .EXTRN  IMPLICIT_DISABLE
                                                        .EXTRN  MIN_SCOPE, MAX_SCOPE
                                                        .EXTRN  OCD_VECTOR

                                        07FC 00000      .ENTRY  FIND_OPERATOR, Save R2,R3,R4,R5,R6,R7,R8,-  ; 0198
                                                                R9,R10                                      ;
                                    08  BC  D4 00002       CLRL    @BLOCK                                  ; 0265
                                        58  D4 00005       CLRL    FOUND                                   ; 0266
                                54 00000000G  8F  D0 00007       MOVL    #MAX_SCOPE, OCD_INDEX               ; 0267
                                56      04    AC  D0 0000E       MOVL    RQCB, R6                            ; 0291
                    00000000G   8F            54  D1 00012 1$:   CMPL    OCD_INDEX, #MIN_SCOPE               ; 0268
                                              54  19 00019       BLSS    7$                                  ;
                                54            58  E8 0001B       BLBS    FOUND, 8$                           ;
                          50    54            01  78 0001E       ASHL    #1, OCD_INDEX, R0                   ; 0273
                                55      0000GCF40   D0 00022       MOVL    OCD_VECTOR-8[R0], OCD               ;
                                5A      0000GCF40   D0 00028       MOVL    OCD_VECTOR-4[R0], OCD_COUNT         ; 0274
                                3A            58  E8 0002E 2$:   BLBS    FOUND, 6$                           ; 0275
                                              38  15 00031       BLEQ    6$                                  ;
                                59      46    A5  3C 00033       MOVZWL  70(OCD), OPER_COUNT                 ; 0280
                                57      50    A5  D0 00037       MOVL    80(OCD), OPER_RQCB                  ; 0281
                                              59  D5 0003B 3$:   TSTL    OPER_COUNT                          ; 0282
                                              25  15 0003D       BLEQ    5$                                  ;
                                22            58  E8 0003F       BLBS    FOUND, 5$                           ;
 7C   A6        00     0080    D7      7C     A7  2D 00042       CMPC5   124(OPER_RQCB), @128(OPER_RQCB), #0, -  ; 0289
                                      0080    D6      0004B               124(R6), @128(R6)                   ;
                                              0D  12 0004E       BNEQ    4$                                  ;
                                58            01  D0 00050       MOVL    #1, FOUND                           ; 0297
                             08 BC            57  D0 00053       MOVL    OPER_RQCB, @BLOCK                   ; 0298
                             24 A6            55  D0 00057       MOVL    OCD, 36(R6)                         ; 0299
                                              DE  11 0005B       BRB     3$                                  ; 0289
                                57            67  D0 0005D 4$:   MOVL    (OPER_RQCB), OPER_RQCB              ; 0303
                                              59  D7 00060       DECL    OPER_COUNT                          ; 0304
                                              D7  11 00062       BRB     3$                                  ; 0282
                                55            65  D0 00064 5$:   MOVL    (OCD), OCD                          ; 0307
                                              5A  D7 00067       DECL    OCD_COUNT                           ; 0308
                                              C3  11 00069       BRB     2$                                  ; 0275
                                              54  D7 0006B 6$:   DECL    OCD_INDEX                           ; 0310
                                              A3  11 0006D       BRB     1$                                  ; 0268
                                0A            58  E9 0006F 7$:   BLBC    FOUND, 9$                           ; 0317
                                              57  DD 00072 8$:   PUSHL   OPER_RQCB                           ; 0319
                             0000G CF         01  FB 00074       CALLS   #1, IMPLICIT_DISABLE                ;
                                58            50  D2 00079       MCOML   R0, FOUND                           ;
                                50            58  D0 0007C 9$:   MOVL    FOUND, R0                           ; 0320
```

```
                                04 0007F        RET                                     ; 0322

; Routine Size:  128 bytes,    Routine Base:  $CODE$ + 0061
```

```
;  326     0323  1 GLOBAL ROUTINE IMPLICIT_DISABLE (OPER_RQCB) =
;  327     0324  1
;  328     0325  1 !++
;  329     0326  1 ! Functional description:
;  330     0327  1 !
;  331     0328  1 !       This routine will determine if an operator device has
;  332     0329  1 !       been implicitly disabled.  That is, if the operator device
;  333     0330  1 !       is no longer marked as an operator.  The OPR bit is cleared
;  334     0331  1 !       when the last channel to a non-allocated device has been
;  335     0332  1 !       released, or when a device is deallocated.  The OPR bit will
;  336     0333  1 !       be reset if the operator is a "permanent" operator.
;  337     0334  1 !
;  338     0335  1 ! Input:
;  339     0336  1 !
;  340     0337  1 !       OPER_RQCB       : Address of an operator RQCB
;  341     0338  1 !
;  342     0339  1 ! Implicit Input:
;  343     0340  1 !
;  344     0341  1 !       None.
;  345     0342  1 !
;  346     0343  1 ! Output:
;  347     0344  1 !
;  348     0345  1 !       None.
;  349     0346  1 !
;  350     0347  1 ! Implict output:
;  351     0348  1 !
;  352     0349  1 !       None.
;  353     0350  1 !
;  354     0351  1 ! Side effects:
;  355     0352  1 !
;  356     0353  1 !       If the operator has been implicitly disabled, and is not
;  357     0354  1 !       a permanent operator, then the operator will be disabled
;  358     0355  1 !       without a disable message being sent to the operator.
;  359     0356  1 !
;  360     0357  1 ! Routine value:
;  361     0358  1 !
;  362     0359  1 !       TRUE    : If the operator is disabled
;  363     0360  1 !       FALSE   : If the operator is still enabled
;  364     0361  1 !--
;  365     0362  1
;  366     0363  2 BEGIN                                   ! Start of IMPLICIT_DISABLE
;  367     0364  2
;  368     0365  2 MAP
;  369     0366  2         OPER_RQCB       : $ref_bblock;  ! Operator RQCB structure
;  370     0367  2
;  371     0368  2 EXTERNAL ROUTINE
;  372     0369  2         CHECK_OPER_COVERAGE,            ! Check coverage for requests
;  373     0370  2         CLUSMSG_RQCB_SEND,              ! Tell the cluster about something
;  374     0371  2         DEALLOCATE_RQCB : NOVALUE,      ! Dispose of an RQCB
;  375     0372  2         UPD_OPER_CONTEXT;               ! Update an operator context
;  376     0373  2
;  377     0374  2 LOCAL
;  378     0375  2         LOST_COVERAGE   : LONG,         ! Boolean
;  379     0376  2         DEV_CHAR        : $bblock [DIB$K_LENGTH],! Device characteristics buffer
;  380     0377  2         CHAR_DESC       : $desc_block,  ! Dev. char. buffer descriptor
;  381     0378  2         OCD             : $ref_bblock,  ! OCD data structure
;  382     0379  2         DISABLED        : LONG,         ! Boolean
```

```
;  383     0380  2          ARG_LIST        : VECTOR [3];               ! Arguement list for EXE$SETOPR
;  384     0381  2
;  385     0382  2      !
;  386     0383  2      ! Do not implicitly disable operators on other nodes
;  387     0384  2      !
;  388     0385  2      IF .GLOBAL_STATUS [GBLSTS_K_IN_VAXcluster]
;  389     0386  2      THEN
;  390     0387  2          IF NOT CLUSUTIL_SYSTEMID_EQUAL (OPER_RQCB [RQCB_T_SYSTEMID], LCL_NOD [NOD_T_NODE_SYSTEMID])
;  391     0388  2          THEN
;  392     0389  2              RETURN FALSE;                           ! Not disabled
;  393     0390  2      !
;  394     0391  2      ! If the operator has a disable in progress, then do not try another one.  This is just in case
;  395     0392  2      ! any routine that we call tries to notify operators.
;  396     0393  2      !
;  397     0394  2      IF .OPER_RQCB [OPRSTS_V_IMPDISABLE]
;  398     0395  2      THEN
;  399     0396  2              RETURN TRUE;
;  400     0397  2      !
;  401     0398  2      ! Create a descriptor for the characteristics buffer and
;  402     0399  2      ! get the operator device characteristics.
;  403     0400  2      !
;  404     0401  2      DISABLED = FALSE;                               ! Assume operator not disabled
;  405     0402  2      CHAR_DESC [0,0,32,0] = DIB$K_LENGTH;            ! Set buffer length
;  406     0403  2      CHAR_DESC [DSC$A_POINTER] = DEV_CHAR;           ! Set buffer address
;  407     0404  3      IF NOT ($GETDEV (DEVNAM=OPER_RQCB [RQCB_L_OPER_LEN], PRIBUF=CHAR_DESC))
;  408     0405  2      THEN
;  409     0406  2          DISABLED = TRUE;                            ! Device no longer exists
;  410     0407  2      !
;  411     0408  2      ! Check the OPR bit.  Reset it if this is a permanent operator.
;  412     0409  2      !
;  413     0410  3      IF NOT (.$bbblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_OPR])
;  414     0411  2      THEN
;  415     0412  2          IF .$bbblock [OPER_RQCB [RQCB_L_RQ_OPTIONS], OPC$V_PERMOPER]
;  416     0413  2          THEN
;  417     0414  3              BEGIN
;  418     0415  3              !
;  419     0416  3              ! Reset the OPR bit in the device UCB.
;  420     0417  3              !
;  421     0418  3              ARG_LIST [0] = 2;                       ! Set number of arguements
;  422     0419  3              ARG_LIST [1] = OPER_RQCB [RQCB_L_OPER_LEN];
;  423     0420  3              ARG_LIST [2] = ON;                      ! Set bit state
;  424     0421  4              IF NOT $CMKRNL (ROUTIN=EXE$SETOPR, ARGLST=ARG_LIST)
;  425     0422  4              THEN
;  426     0423  4                  DISABLED = TRUE;
;  427     0424  3              END
;  428     0425  2          ELSE
;  429     0426  2              DISABLED = TRUE;
;  430     0427  2      !
;  431     0428  2      ! If the operator is disabled, then remove it from the operator list.
;  432     0429  2      ! Do not notify the operator of the disable.  After doing the disable,
;  433     0430  2      ! check to see if any requests have lost operator coverage.
;  434     0431  2      !
;  435     0432  2      IF .DISABLED
;  436     0433  2      THEN
;  437     0434  3          BEGIN
;  438     0435  3          !
;  439     0436  3          ! The rqcb is tainted, let everybody know
```

```
;  440          0437  3      !
;  441          0438  3      OPER_RQCB [OPRSTS_V_IMPDISABLE] = TRUE;
;  442          0439  3      !
;  443          0440  3      ! Tell the cluster to disable this operator
;  444          0441  3      !
;  445          0442  3      CLUSMSG_RQCB_SEND (-1, CLM_IMP_DISABLE, .OPER_RQCB);
;  446          0443  3      LOST_COVERAGE = UPD_OPER_CONTEXT    (TRUE,                      ! Do the disable
;  447          0444  3                                          .OPER_RQCB [RQCB_L_ATTNMASK1],
;  448          0445  3                                          .OPER_RQCB [RQCB_L_ATTNMASK2],
;  449          0446  3                                          .OPER_RQCB
;  450          0447  3                                         );
;  451          0448  3      REMQUE (.OPER_RQCB, OPER_RQCB);              ! Remove RQCB from operator list
;  452          0449  3      OCD = .OPER_RQCB [RQCB_L_OCD];               ! Get OCD address
;  453          0450  3      OCD [OCD_W_OPERCOUNT] = .OCD [OCD_W_OPERCOUNT] - 1;
;  454          0451  3      DEALLOCATE_RQCB (.OPER_RQCB);                ! Dispose of the RQCB
;  455          0452  3      !
;  456          0453  3      ! If operator coverage was lost due to the disable, check all
;  457          0454  3      ! outstanding reuquests queued to this OCD for operator coverage.
;  458          0455  3      ! All requests that no longer have operator coverage will be canceled.
;  459          0456  3      !
;  460          0457  3      IF .LOST_COVERAGE
;  461          0458  3      THEN
;  462          0459  3          CHECK_OPER_COVERAGE (.OCD);
;  463          0460  3      END;
;  464          0461  3
;  465          0462  2  RETURN (.DISABLED);                              ! Return the routine value
;  466          0463  2
;  467          0464  1  END;                                             ! End of IMPLICIT_DISABLE


                                                .EXTRN  CLUSMSG_RQCB_SEND
                                                .EXTRN  UPD_OPER_CONTEXT
                                                .EXTRN  SYS$GETDEV, EXE$SETOPR
                                                .EXTRN  SYS$CMKRNL

                                    001C 00000          .ENTRY  IMPLICIT_DISABLE, Save R2,R3,R4                 ; 0323
                          5E      FF78   CE   9E 00002  MOVAB   -136(SP), SP                                    ;
                          18      0000G  CF   E9 00007  BLBC    GLOBAL_STATUS+1, 1$                             ; 0385
51          0000G   CF 00000050       8F   C1 0000C  ADDL3   #80, LCL_NOD, R1                                ; 0387
50            04    AC                1C   C1 00016  ADDL3   #28, OPER_RQCB, R0
                                    0000G 30 0001B  BSBW    CLUSUTIL_SYSTEMID_EQUAL
                          03              50   E8 0001E  BLBS    R0, 1$
                                    00A2  31 00021  BRW     7$
                          52        04    AC   D0 00024 1$:  MOVL    OPER_RQCB, R2                               ; 0394
04          78    A2                03   E1 00028  BBC     #3, T20(R2), 2$
                          50              01   D0 0002D  MOVL    #1, R0                                      ; 0396
                                          04   00030  RET
                                          53   D4 00031 2$:  CLRL    DISABLED                                    ; 0401
                    0C    AE        74    8F   9A 00033  MOVZBL  #116, CHAR_DESC                              ; 0402
                    10    AE        14    AE   9E 00038  MOVAB   DEV_CHAR, CHAR_DESC+4                        ; 0403
                                    7E    7C   0003D  CLRQ    -(SP)                                           ; 0404
                                    14    AE   9F 0003F  PUSHAB  CHAR_DESC
                                    7E    D4   00042  CLRL    -(SP)
                                    7C    A2   9F 00044  PUSHAB  124(R2)
                00000000G   00              05   FB 00047  CALLS   #5, SYS$GETDEV
                          03              50   E8 0004E  BLBS    R0, 3$                                      ;
```

```
                    53          01 D0 00051          MOVL    #1, DISABLED                    ; 0406
                    14      AE  95 00054 3$:         TSTB    DEV_CHAR                        ; 0410
                                19 00057             BLSS    5$
                1E  58      A2  01 E1 00059          BBC     #1, 88(R2), 4$                  ; 0412
                            6E  02 D0 0005E          MOVL    #2, ARG_LIST                    ; 0418
                04  AE      7C  A2 9E 00061          MOVAB   124(R2), ARG_LIST+4             ; 0419
                08  AE          01 D0 00066          MOVL    #1, ARG_LIST+8                  ; 0420
                                5E DD 0006A          PUSHL   SP                              ; 0421
                        00000000G   00 9F 0006C      PUSHAB  EXE$SETOPR
            00000000G   00          02 FB 00072      CALLS   #2, SYS$CMKRNL
                        03          50 E8 00079      BLBS    R0, 5$
                        53          01 D0 0007C 4$:  MOVL    #1, DISABLED                    ; 0426
                        40          53 E9 0007F 5$:  BLBC    DISABLED, 6$                    ; 0432
                78  A2              08 88 00082      BISB2   #8, 120(R2)                     ; 0438
                                    52 DD 00086      PUSHL   R2                              ; 0442
                                    0A DD 00088      PUSHL   #10
                    7E              01 CE 0008A      MNEGL   #1, -(SP)
                0000G   CF          03 FB 0008D      CALLS   #3, CLUSMSG_RQCB_SEND
                                    52 DD 00092      PUSHL   R2                              ; 0446
                    7E      5C  A2  7D 00094         MOVQ    92(R2), -(SP)                   ; 0444
                                    01 DD 00098      PUSHL   #1                              ; 0443
                0000G   CF          04 FB 0009A      CALLS   #4, UPD_OPER_CONTEXT
                    54              50 D0 0009F      MOVL    R0, LOST_COVERAGE
                04  AC              62 0F 000A2      REMQUE  (R2), OPER_RQCB                 ; 0448
                    50      04  AC  D0 000A6         MOVL    OPER_RQCB, R0                   ; 0449
                    52      24  A0  D0 000AA         MOVL    36(R0), OCD
                            46  A2  B7 000AE         DECW    70(OCD)                         ; 0450
                                    50 DD 000B1      PUSHL   R0                              ; 0451
                0000G   CF          01 FB 000B3      CALLS   #1, DEALLOCATE_RQCB
                    07              54 E9 000B8      BLBC    LOST_COVERAGE, 6$               ; 0457
                                    52 DD 000BB      PUSHL   OCD                             ; 0459
                0000G   CF          01 FB 000BD      CALLS   #1, CHECK_OPER_COVERAGE
                    50              53 D0 000C2 6$:  MOVL    DISABLED, R0                    ; 0462
                                    04 000C5         RET
                                50  D4 000C6 7$:     CLRL    R0                              ; 0464
                                    04 000C8         RET

; Routine Size:  201 bytes,    Routine Base:  $CODE$ + 00E1
```

```
;  469     0465  1 GLOBAL ROUTINE NOTIFY_LISTED_OPERATORS (RQST_RQCB) =
;  470     0466  1
;  471     0467  1 !++
;  472     0468  1 ! Functional description:
;  473     0469  1 !
;  474     0470  1 !       This routine will traverse a list of operators
;  475     0471  1 !       (pointed to by the OCD pointed to by the request RQCB)
;  476     0472  1 !       and send the message associated with the RQCB to all
;  477     0473  1 !       operators who are enabled to receive the message.
;  478     0474  1 !
;  479     0475  1 ! Input:
;  480     0476  1 !
;  481     0477  1 !       RQST_RQCB       : Address of a request RQCB
;  482     0478  1 !
;  483     0479  1 ! Implicit Input:
;  484     0480  1 !
;  485     0481  1 !       None.
;  486     0482  1 !
;  487     0483  1 ! Output:
;  488     0484  1 !
;  489     0485  1 !       None.
;  490     0486  1 !
;  491     0487  1 ! Implict output:
;  492     0488  1 !
;  493     0489  1 !       The message will be sent to the interested operators.
;  494     0490  1 !
;  495     0491  1 ! Side effects:
;  496     0492  1 !
;  497     0493  1 !       As part of sending the message, the operators are checked
;  498     0494  1 !       to see if they have been implicitly disabled.  If so, they
;  499     0495  1 !       are removed from the operator list.
;  500     0496  1 !
;  501     0497  1 ! Routine value:
;  502     0498  1 !
;  503     0499  1 !       TRUE    : If at least one operator was notified.
;  504     0500  1 !       FALSE   : If no operators were notified.
;  505     0501  1 !--
;  506     0502  1
;  507     0503  2 BEGIN                                           ! Start of NOTIFY_LISTED_OPERATORS
;  508     0504  2
;  509     0505  2 MAP
;  510     0506  2         RQST_RQCB       : $ref_bblock;          ! Request RQCB structure
;  511     0507  2
;  512     0508  2 EXTERNAL ROUTINE
;  513     0509  2         IMPLICIT_DISABLE,                       ! Check for implicit disable
;  514     0510  2         NOTIFY_OPERATOR;                        ! Send a message to a given operator
;  515     0511  2
;  516     0512  2 LOCAL
;  517     0513  2         OCD             : $ref_bblock,          ! OCD data structure
;  518     0514  2         SAVED_MCB       : LONG,                 ! Address of an MCB
;  519     0515  2         OPER_COUNT      : LONG,                 ! Count of operators on list
;  520     0516  2         CURRENT_OPER    : $ref_bblock,          ! Current operator RQCB
;  521     0517  2         NEXT_OPER       : $ref_bblock,          ! Next operator RQCB
;  522     0518  2         NOTIFIED        : LONG;                 ! Boolean
;  523     0519  2
;  524     0520  2 NOTIFIED = FALSE;                               ! Assume no operator notified
;  525     0521  2 !
```

```
;  526     0522  2 ! Check the request to see if NOBRD is specified.  If it is, and the requestor
;  527     0523  2 ! has the proper privileges, return failure without sending the message.
;  528     0524  2 !
;  529     0525  2 IF .$bblock [RQST_RQCB [RQCB_L_OPTIONS], OPC$V_NOBRD]
;  530     0526  2 THEN
;  531     0527  2     IF (.$bblock [RQST_RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER])
;  532     0528  2     OR ((.RQST_RQCB [RQCB_B_SCOPE] EQL OPC$K_GROUP) AND ((.$bblock [RQST_RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER
;  533     0529  2                                                           (.$bblock [RQST_RQCB [RQCB_L_PRIVMASK1], PRV$V_GROU
;  534     0530  2 THEN
;  535     0531  2     RETURN (FALSE);
;  536     0532  2 OCD = .RQST_RQCB [RQCB_L_OCD];                  ! Get OCD address
;  537     0533  2 IF .OCD EQL 0
;  538     0534  2 THEN
;  539     0535  2     BEGIN
;  540     0536  2     LOCAL
;  541     0537  2         DESC : VECTOR [2, LONG];
;  542     0538  2     DESC [0] = RQCB_K_SIZE;
;  543     0539  2     DESC [1] = .RQST_RQCB;
;  544     0540  2     DUMP_LOG_FILE (DESC, %ASCID 'OCD address is zero in RQCB');
;  545     0541  2     RETURN (FALSE);
;  546     0542  2     END;
;  547     0543  2 OPER_COUNT = .OCD [OCD_W_OPERCOUNT];            ! Get count of operators
;  548     0544  2 NEXT_OPER = .OCD [OCD_L_OPERFLINK];             ! Get address of next operator in list
;  549     0545  2 WHILE (.OPER_COUNT GTR 0) DO
;  550     0546  2     BEGIN
;  551     0547  3     !
;  552     0548  3     ! Link to the next operator RQCB.  We have to keep the address
;  553     0549  3     ! of the next operator RQCB in case this one evaporates as a
;  554     0550  3     ! side effect of IMPLICT_DISABLE.
;  555     0551  3     !
;  556     0552  3     CURRENT_OPER = .NEXT_OPER;
;  557     0553  3     NEXT_OPER = .CURRENT_OPER [RQCB_L_FLINK];
;  558     0554  3     !
;  559     0555  3     ! Check the request attention mask against the operator's
;  560     0556  3     ! enable mask.  If an bits in common, then notify the operator.
;  561     0557  3     ! The message is also sent if a special status bit is set.
;  562     0558  3     ! This is an internal hack used to force message output.
;  563     0559  3     !
;  564     0560  4     IF ((.RQST_RQCB [RQCB_L_ATTNMASK1] AND .CURRENT_OPER [RQCB_L_ATTNMASK1]) NEQ 0)
;  565     0561  4     OR ((.RQST_RQCB [RQCB_L_ATTNMASK2] AND .CURRENT_OPER [RQCB_L_ATTNMASK2]) NEQ 0)
;  566     0562  4     OR (.RQST_RQCB [HDR_V_BRD])
;  567     0563  3     THEN
;  568     0564  4         IF NOT (IMPLICIT_DISABLE (.CURRENT_OPER))
;  569     0565  3         THEN
;  570     0566  4             BEGIN
;  571     0567  4             !
;  572     0568  4             ! Send the message to the operator.  The MCB from the RQST_RQCB is
;  573     0569  4             ! reused to avoid the overhead of creating a new MCB for each operator.
;  574     0570  4             !
;  575     0571  4             SAVED_MCB = .CURRENT_OPER [RQCB_L_MCB];
;  576     0572  4             CURRENT_OPER [RQCB_L_MCB] = .RQST_RQCB [RQCB_L_MCB];
;  577     0573  4             IF NOTIFY_OPERATOR (.CURRENT_OPER)
;  578     0574  4             THEN
;  579     0575  4                 NOTIFIED = TRUE;                ! An operator was notified
;  580     0576  4             CURRENT_OPER [RQCB_L_MCB] = .SAVED_MCB;
;  581     0577  3             END;
;  582     0578  3     OPER_COUNT = .OPER_COUNT - 1;               ! Decrement the operator count
```

```
;  583          0579  2     END;
;  584          0580  2
;  585          0581  2 RETURN (.NOTIFIED);                    ! Return routine value
;  586          0582  2
;  587          0583  1 END;                                    ! End of NOTIFY_LISTED_OPERATORS


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

20 73 69 20 73 73 65 72 64 64 61 20 44 43 4F  00000 P.AAB:    .ASCII  \OCD address is zero in RQCB\<0>                    ;
   00 42 43 51 52 20 6E 69 20 6F 72 65 7A      0000F
                                  010E001B     0001C P.AAA:    .LONG   17694747                                           ;
                                  00000000'    00020           .ADDRESS P.AAB                                             ;

                                                                .EXTRN  NOTIFY_OPERATOR

                                                                .PSECT  $CODE$,NOWRT,2

                                       00FC 00000               .ENTRY  NOTIFY_LISTED_OPERATORS, Save R2,R3,R4,R5,-     ; 0465
                                                                R6,R7                                                      ;
                                 5E            08  C2 00002   SUBL2   #8, SP                                            ;
                                               57  D4 00005   CLRL    NOTIFIED                                           ; 0520
                                 53        04  AC  D0 00007   MOVL    RQST_RQCB, R3                                      ; 0525
                           14         54  A3      01  E1 0000B   BBC     #1, 84(R3), 1$                                     ;
                           7C         32  A3      02  E0 00010   BBS     #2, 50(R3), 8$                                     ; 0527
                                 02        53  A3  91  00015   CMPB    83(R3), #2                                         ; 0528
                                               09  12 00019   BNEQ    1$                                                 ;
                           71         32  A3      02  E0 0001B   BBS     #2, 50(R3), 8$                                     ;
                                 6D        31  A3  E8  00020   BLBS    49(R3), 8$                                         ; 0529
                                 52        24  A3  D0  00024 1$: MOVL    36(R3), OCD                                        ; 0532
                                               16  12 00028   BNEQ    2$                                                 ; 0533
                                 6E        94  8F  9A  0002A   MOVZBL  #148, DESC                                          ; 0538
                              04 AE            53  D0  0002E   MOVL    R3, DESC+4                                         ; 0539
                                       0000'   CF  9F  00032   PUSHAB  P.AAA                                              ; 0540
                                          04   AE  9F  00036   PUSHAB  DESC                                               ;
                         0000G CF              02  FB  00039   CALLS   #2, DUMP_LOG_FILE                                 ;
                                               51  11  0003E   BRB     8$                                                 ; 0541
                                 55        46  A2  3C  00040 2$: MOVZWL  70(OCD), OPER_COUNT                               ; 0543
                                 54        50  A2  D0  00044   MOVL    80(OCD), NEXT_OPER                                 ; 0544
                                               55  D5  00048 3$: TSTL    OPER_COUNT                                         ; 0545
                                               41  15  0004A   BLEQ    7$                                                 ;
                                 52            54  D0  0004C   MOVL    NEXT_OPER, CURRENT_OPER                            ; 0552
                                 54            62  D0  0004F   MOVL    (CURRENT_OPER), NEXT_OPER                          ; 0553
                              5C A2        5C  A3  D3  00052   BITL    92(R3), 92(CURRENT_OPER)                           ; 0560
                                               0C  12 00057   BNEQ    4$                                                 ;
                              60 A2        60  A3  D3  00059   BITL    96(R3), 96(CURRENT_OPER)                           ; 0561
                                               05  12 0005E   BNEQ    4$                                                 ;
                           24 28 A3            01  E1  00060   BBC     #1, 40(R3), 6$                                     ; 0562
                                               52  DD  00065 4$: PUSHL   CURRENT_OPER                                     ; 0564
                         0000G CF              01  FB  00067   CALLS   #1, IMPLICIT_DISABLE                              ;
                                 1A            50  E8  0006C   BLBS    R0, 6$                                             ;
                                 56        6C  A2  D0  0006F   MOVL    108(CURRENT_OPER), SAVED_MCB                       ; 0571
                              6C A2        6C  A3  D0  00073   MOVL    108(R3), 108(CURRENT_OPER)                         ; 0572
                                               52  DD  00078   PUSHL   CURRENT_OPER                                     ; 0573
                         0000G CF              01  FB  0007A   CALLS   #1, NOTIFY_OPERATOR                               ;
                                 03            50  E9  0007F   BLBC    R0, 5$                                             ;
```

```
                     57          01  D0 00082          MOVL    #1, NOTIFIED                                    ; 0575
                 6C  A2          56  D0 00085 5$:       MOVL    SAVED_MCB, 108(CURRENT_OPER)                    ; 0576
                                 55  D7 00089 6$:       DECL    OPER_COUNT                                      ; 0578
                                 BB  11 0008B           BRB     3$                                              ; 0545
                     50          57  D0 0008D 7$:       MOVL    NOTIFIED, R0                                    ; 0581
                                     04 00090           RET                                                     ;
                                 50  D4 00091 8$:       CLRL    R0                                              ; 0583
                                     04 00093           RET                                                     ;

; Routine Size:  148 bytes,    Routine Base:  $CODE$ + 01AA
```

```
;  589     0584  1 GLOBAL ROUTINE NOTIFY_OPERATOR (RQCB) =
;  590     0585  1
;  591     0586  1 !++
;  592     0587  1 ! Functional description:
;  593     0588  1 !
;  594     0589  1 !       This routine will send a message to an operator,
;  595     0590  1 !       be it a terminal or a mailbox.
;  596     0591  1 !
;  597     0592  1 ! Input:
;  598     0593  1 !
;  599     0594  1 !       RQCB            : Address of an operator RQCB
;  600     0595  1 !
;  601     0596  1 ! Implicit Input:
;  602     0597  1 !
;  603     0598  1 !       The RQCB points to an MCB that describes the message.
;  604     0599  1 !
;  605     0600  1 ! Output:
;  606     0601  1 !
;  607     0602  1 !       None.
;  608     0603  1 !
;  609     0604  1 ! Implict output:
;  610     0605  1 !
;  611     0606  1 !       A message is sent to the operator.
;  612     0607  1 !
;  613     0608  1 ! Side effects:
;  614     0609  1 !
;  615     0610  1 !       If the operator device is a mailbox, the message
;  616     0611  1 !       may be truncated if the mailbox buffer size is not
;  617     0612  1 !       large enough to hold the entire message.
;  618     0613  1 !
;  619     0614  1 ! Routine value:
;  620     0615  1 !
;  621     0616  1 !       TRUE            : If success
;  622     0617  1 !       <anything else> : If the message could not be sent
;  623     0618  1 !--
;  624     0619  2 BEGIN                                           ! Start of NOTIFY_OPERATOR
;  625     0620  2
;  626     0621  2 MAP
;  627     0622  2         RQCB            : $ref_bblock;          ! Operator RQCB structure
;  628     0623  2
;  629     0624  2 LOCAL
;  630     0625  2         OCD             : $ref_bblock,          ! OCD data structure
;  631     0626  2         MSG_SIZE        : WORD,                 ! Size of message to operator
;  632     0627  2         MBX_CHANNEL     : WORD,                 ! Channel to operator mailbox
;  633     0628  2         IOSB            : $bblock [8],          ! I/O status block
;  634     0629  2         MCB             : $ref_bblock,          ! MCB data structure
;  635     0630  2         STATUS          : LONG;
;  636     0631  2
;  637     0632  2 !
;  638     0633  2 ! If there is no MCB connected to the RQCB, then return an error status.
;  639     0634  2 !
;  640     0635  2 MCB = .RQCB [RQCB_L_MCB];
;  641     0636  2 IF .MCB EQL 0
;  642     0637  2 THEN
;  643     0638  2     RETURN (FALSE);
;  644     0639  2 !
;  645     0640  2 ! Check the request to see if NOBRD is specified.  If it is, and the requestor
```

```
;  646     0641  2 ! has the proper privileges, return failure without sending the message.
;  647     0642  2 !
;  648     0643  2 IF .$bblock [RQCB [RQCB_L_OPTIONS], OPC$V_NOBRD]
;  649     0644  2 THEN
;  650     0645  3     IF (.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER])
;  651     0646  4     OR ((.RQCB [RQCB_B_SCOPE] EQL OPC$K_GROUP) AND ((.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER]) OR
;  652     0647  3                                                     (.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_GROUP])))
;  653     0648  2 THEN
;  654     0649  2     RETURN (FALSE);
;  655     0650  2 !
;  656     0651  2 ! If the operator is on another node, then pretend that we notified the operator.  OPCOMRQST uses the
;  657     0652  2 ! value to determine if a request can be fielded.
;  658     0653  2 !
;  659     0654  2 IF .GLOBAL_STATUS [GBLSTS_K_IN_VAXcluster]
;  660     0655  2 THEN
;  661     0656  2     IF NOT CLUSUTIL_SYSTEMID_EQUAL (RQCB [RQCB_T_SYSTEMID], LCL_NOD [NOD_T_NODE_SYSTEMID])
;  662     0657  2     THEN
;  663     0658  2         RETURN (TRUE);
;  664     0659  2 !
;  665     0660  2 ! Send the message.  How it is sent depends on the device type.
;  666     0661  2 !
;  667     0662  2 IF .RQCB [OPRSTS_V_TRM]
;  668     0663  2 OR .RQCB [OPRSTS_V_REMTRM]
;  669     0664  2 THEN
;  670     0665  3     BEGIN
;  671     0666  3     !
;  672     0667  3     ! The operator device is a terminal or remote terminal.
;  673     0668  3     ! Send the message via $BRKTHRU
;  674     0669  3     !
;  675     0670  3     REPLYBRD_BRKTHRU_QUEUE (
;  676     0671  3                             MCB [MCB_L_TEXTLEN],    ! Message to send
;  677     0672  3                             RQCB [RQCB_L_OPER_LEN], ! Target (operator device name)
;  678     0673  3                             BRK$C_DEVICE,           ! Type of target
;  679     0674  3                             32,                     ! Carriage control
;  680     0675  3                             0,                      ! Flags
;  681     0676  3                             BRK$C_OPCOM,            ! Type of requestor
;  682     0677  3                             0,0,0,0,0);             ! No completion routine or arguments
;  683     0678  3     RETURN 1;
;  684     0679  3     END
;  685     0680  2 ELSE
;  686     0681  2     IF .RQCB [OPRSTS_V_MBX]
;  687     0682  2     THEN
;  688     0683  3         BEGIN
;  689     0684  3         !
;  690     0685  3         ! The operator device is a mailbox.
;  691     0686  3         ! Send the message via $QIO.  If the mailbox is
;  692     0687  3         ! too small, truncate the message to fit.
;  693     0688  3         !
;  694     0689  3         MSG_SIZE = .MCB [MCB_L_TEXTLEN];                ! Assume mailbox big enough
;  695     0690  3         IF .MSG_SIZE GTR .RQCB [RQCB_W_MBXSIZE]         ! Is message to big?
;  696     0691  3         THEN
;  697     0692  3             MSG_SIZE = .RQCB [RQCB_W_MBXSIZE];          ! Yes, truncate message
;  698     0693  3
;  699   P 0694  4         IF NOT (STATUS = $ASSIGN (CHAN = MBX_CHANNEL,   ! Assign a channel to the  operator device
;  700   P 0695  4                                  DEVNAM = RQCB [RQCB_L_OPER_LEN]
;  701     0696  4                                 ))
;  702     0697  3         THEN
```

```
;  703          0698  3              RETURN (.STATUS);
;  704          0699  3
;  705        P 0700  4          IF (STATUS = $QIOW (FUNC = (IO$_WRITEVBLK OR IO$M_NOW), ! Send the message
;  706        P 0701  4                              CHAN = .MBX_CHANNEL,
;  707        P 0702  4                              IOSB = IOSB,
;  708        P 0703  4                              P1   = .MCB [MCB_L_TEXTPTR],
;  709        P 0704  4                              P2   = .MSG_SIZE
;  710          0705  4                             ))
;  711          0706  3          THEN
;  712          0707  3              STATUS = .IOSB [0,0,16,0];          ! Get actual I/O operation status
;  713          0708  3
;  714          0709  3          $DASSGN (CHAN = .MBX_CHANNEL);          ! Deassign channel to operator device
;  715          0710  3          RETURN (.STATUS);                       ! Return the appropriate status
;  716          0711  2          END;
;  717          0712  2
;  718          0713  2      ! If we get this far, it means that the device is not a
;  719          0714  2      ! legal operator device, and that the message cannot be sent.
;  720          0715  2      ! Return an error status.
;  721          0716  2      !
;  722          0717  2      RETURN (FALSE);
;  723          0718  2
;  724          0719  1  END;                                            ! End of NOTIFY_OPERATOR


                                                .EXTRN  SYS$ASSIGN, SYS$QIOW
                                                .EXTRN  SYS$DASSGN

                                001C 00000      .ENTRY  NOTIFY_OPERATOR, Save R2,R3,R4          ; 0584
                          5E        0C  C2 00002      SUBL2   #12, SP                         ;
                          52    04  AC  D0 00005      MOVL    RQCB, R2                        ; 0635
                          53    6C  A2  D0 00009      MOVL    108(R2), MCB                    ;
                                    03  12 0000D      BNEQ    2$                              ; 0636
                                00B6  31 0000F  1$:   BRW     11$                             ;
                14        54  A2      01  E1 00012  2$:   BBC     #1, 84(R2), 3$                  ; 0643
                F3        32  A2      02  E0 00017        BBS     #2, 50(R2), 1$                  ; 0645
                          02    53  A2  91 0001C        CMPB    83(R2), #2                      ; 0646
                                    09  12 00020        BNEQ    3$                              ;
                E8        32  A2      02  E0 00022        BBS     #2, 50(R2), 1$                  ;
                          E4    31  A2  E8 00027        BLBS    49(R2), 1$                      ; 0647
                          14  0000G CF  E9 0002B  3$:   BLBC    GLOBAL_STATUS+1, 4$             ; 0654
                51  0000G  CF 00000050 8F  C1 00030        ADDL3   #80, LCL_NOD, R1                ; 0656
                          50    1C  A2  9E 0003A        MOVAB   28(R2), R0                      ;
                                0000G 30 0003E        BSBW    CLUSUTIL_SYSTEMID_EQUAL         ;
                          20        50  E9 00041        BLBC    R0, 6$                          ;
                          05    78  A2  E8 00044  4$:   BLBS    120(R2), 5$                     ; 0662
                1B        78  A2      01  E1 00048        BBC     #1, 120(R2), 7$                 ; 0663
                                  7E  7C 0004D  5$:   CLRQ    -(SP)                           ; 0672
                                  7E  7C 0004F        CLRQ    -(SP)                           ;
                          7E        07  7D 00051        MOVQ    #7, -(SP)                       ;
                          7E        20  7D 00054        MOVQ    #32, -(SP)                      ;
                                    01  DD 00057        PUSHL   #1                              ;
                                7C  A2  9F 00059        PUSHAB  124(R2)                         ;
                                30  A3  9F 0005C        PUSHAB  48(MCB)                         ; 0671
                      0000G     CF      0B  FB 0005F        CALLS   #11, REPLYBRD_BRKTHRU_QUEUE     ; 0672
                          50            01  D0 00064  6$:   MOVL    #1, R0                      ; 0678
                                        04  00067        RET                                     ;
```

```
                5B      78  A2              02  E1  00068 7$:     BBC     #2, 120(R2), 11$                ;0681
                        54      30      A3  B0  0006D           MOVW    48(MCB), MSG_SIZE               ;0689
                    7A  A2              54  B1  00071           CMPW    MSG_SIZE, 122(R2)               ;0690
                                        04  1B  00075           BLEQU   8$
                        54      7A      A2  B0  00077           MOVW    122(R2), MSG_SIZE               ;0692
                                        7E  7C  0007B 8$:       CLRQ    -(SP)                           ;0696
                                    08  AE  9F  0007D           PUSHAB  MBX_CHANNEL
                                    7C  A2  9F  00080           PUSHAB  124(R2)
                00000000G           00      04  FB  00083       CALLS   #4, SYS$ASSIGN
                                            50  D0  0008A       MOVL    R0, STATUS
                                            52  E9  0008D       BLBC    STATUS, 10$
                                            7E  7C  00090       CLRQ    -(SP)                           ;0705
                                            7E  7C  00092       CLRQ    -(SP)
                        7E                  54  3C  00094       MOVZWL  MSG_SIZE, -(SP)
                                        34  A3  DD  00097       PUSHL   52(MCB)
                                            7E  7C  0009A       CLRQ    -(SP)
                                        24  AE  9F  0009C       PUSHAB  IOSB
                        7E              70  8F  9A  0009F       MOVZBL  #112, -(SP)
                        7E              28  AE  3C  000A3       MOVZWL  MBX_CHANNEL, -(SP)
                                            7E  D4  000A7       CLRL    -(SP)
                00000000G           00      0C  FB  000A9       CALLS   #12, SYS$QIOW
                                            50  D0  000B0       MOVL    R0, STATUS
                                            52  E9  000B3       BLBC    STATUS, 9$
                        52          04  AE  3C  000B6           MOVZWL  IOSB, STATUS                    ;0707
                        7E              6E  3C  000BA 9$:       MOVZWL  MBX_CHANNEL, -(SP)              ;0709
                00000000G           00      01  FB  000BD       CALLS   #1, SYS$DASSGN
                        50                  52  D0  000C4 10$:  MOVL    STATUS, R0                      ;0710
                                            04  000C7          RET
                                            50  D4  000C8 11$:  CLRL    R0                              ;0719
                                            04  000CA          RET                                     ;

; Routine Size:  203 bytes,    Routine Base:  $CODE$ + 023E
```

```
;  726     0720  1 GLOBAL ROUTINE OPERUTIL_CLM_IMP_DISABLE (BUFFER_DESC : $ref_bblock, CLM : $ref_bblock, LEN) : NOVALUE =
;  727     0721  1
;  728     0722  1 !++
;  729     0723  1 ! Functional description:
;  730     0724  1 !
;  731     0725  1 !       This routine processes an implicit disable request from another node.
;  732     0726  1 !
;  733     0727  1 ! Input:
;  734     0728  1 !
;  735     0729  1 !       BUFFER_DESC -   pointer to message from remote node, including $SNDOPR header
;  736     0730  1 !       CLM -           pointer to CLMRQCB structure
;  737     0731  1 !       LEN -           length of LEN
;  738     0732  1 !
;  739     0733  1 ! Implicit Input:
;  740     0734  1 !
;  741     0735  1 !       None.
;  742     0736  1 !
;  743     0737  1 ! Output:
;  744     0738  1 !
;  745     0739  1 !       None.
;  746     0740  1 !
;  747     0741  1 ! Implict output:
;  748     0742  1 !
;  749     0743  1 !       None.
;  750     0744  1 !
;  751     0745  1 ! Side effects:
;  752     0746  1 !
;  753     0747  1 !       If the operator has been implicitly disabled, and is not
;  754     0748  1 !       a permanent operator, then the operator will be disabled
;  755     0749  1 !       without a disable message being sent to the operator.
;  756     0750  1 !
;  757     0751  1 ! Routine value:
;  758     0752  1 !
;  759     0753  1 !       TRUE    : If the operator is disabled
;  760     0754  1 !       FALSE   : If the operator is still enabled
;  761     0755  1 !--
;  762     0756  1
;  763     0757  2 BEGIN                                           ! Start of OPERUTIL_CLM_IMP_DISABLE
;  764     0758  2
;  765     0759  2 EXTERNAL ROUTINE
;  766     0760  2         CLUSMSG_CONV_CLM_RQCB,                  ! Convert message to RQCB
;  767     0761  2         CHECK_OPER_COVERAGE,                    ! Check coverage for requests
;  768     0762  2         DEALLOCATE_RQCB : NOVALUE,              ! Dispose of an RQCB
;  769     0763  2         DUMP_LOG_FILE,                          ! Place random string in log
;  770     0764  2         UPD_OPER_CONTEXT;                       ! Update an operator context
;  771     0765  2
;  772     0766  2 LOCAL
;  773     0767  2         FOUND           : LONG,                 ! Found the operator
;  774     0768  2         LOST_COVERAGE   : LONG,                 ! Boolean
;  775     0769  2         DEV_CHAR        : $bblock [DIB$K_LENGTH],! Device characteristics buffer
;  776     0770  2         CHAR_DESC       : $desc_block,          ! Dev. char. buffer descriptor
;  777     0771  2         OCD             : $ref_bblock,          ! OCD data structure
;  778     0772  2         OPER_RQCB       : $ref_bblock,          ! RQCB data structure
;  779     0773  2         RQCB            : $ref_bblock,          ! RQCB data structure
;  780     0774  2         DISABLED        : LONG,                 ! Boolean
;  781     0775  2         ARG_LIST        : VECTOR [3];           ! Arguement list for EXE$SETOPR
;  782     0776  2
```

```
;  783          0777  2 !
;  784          0778  2 ! Check the version number of the message.  If the message is from any other version,
;  785          0779  2 ! simply ignore it.
;  786          0780  2 !
;  787          0781  2 IF .CLM [CLM_B_DS_VERSION] NEQ CLMRQCB_K_DS_VERSION
;  788          0782  2 THEN
;  789          0783  2     RETURN DUMP_LOG_FILE (.BUFFER_DESC, %ASCID 'CLM__OPRENABLE mismatch');
;  790          0784  2 !
;  791          0785  2 ! Allocate an RQCB and convert the message RQCB into the new RQCB
;  792          0786  2 !
;  793          0787  2 IF NOT CLUSMSG_CONV_CLM_RQCB (.CLM, RQCB)
;  794          0788  2 THEN
;  795          0789  2     RETURN DUMP_LOG_FILE (.BUFFER_DESC, ascid_INVALIDRQCB);
;  796          0790  2 !
;  797          0791  2 ! See if the operator is already known to OPCOM.  This entails scanning down the appropriate operator
;  798          0792  2 ! list and comparing the device names for equality.  FIND_OPERATOR will set RQCB [RQCB_L_OCD] if it
;  799          0793  2 ! finds a match.
;  800          0794  2 !
;  801          0795  2 FOUND = FIND_OPERATOR (.RQCB, OPER_RQCB);
;  802          0796  2 DEALLOCATE_RQCB (.RQCB);                          ! Don't need this any more
;  803          0797  2 !
;  804          0798  2 ! The operator has been disabled on the remote node, remove it from the operator list.
;  805          0799  2 ! Do not notify anyone of the disable.  After doing the disable,
;  806          0800  2 ! check to see if any requests have lost operator coverage.
;  807          0801  2 !
;  808          0802  2 IF .FOUND
;  809          0803  2 THEN
;  810          0804  3     BEGIN
;  811          0805  3     LOST_COVERAGE = UPD_OPER_CONTEXT    (TRUE,                  ! Do the disable
;  812          0806  3                                          .OPER_RQCB [RQCB_L_ATTNMASK1],
;  813          0807  3                                          .OPER_RQCB [RQCB_L_ATTNMASK2],
;  814          0808  3                                          .OPER_RQCB
;  815          0809  3                                         );
;  816          0810  3     REMQUE (.OPER_RQCB, OPER_RQCB);              ! Remove RQCB from operator list
;  817          0811  3     OCD = .OPER_RQCB [RQCB_L_OCD];               ! Get OCD address
;  818          0812  3     OCD [OCD_W_OPERCOUNT] = .OCD [OCD_W_OPERCOUNT] - 1;
;  819          0813  3     DEALLOCATE_RQCB (.OPER_RQCB);                ! Dispose of the RQCB
;  820          0814  3     !
;  821          0815  3     ! If operator coverage was lost due to the disable, check all
;  822          0816  3     ! outstanding reuquests queued to this OCD for operator coverage.
;  823          0817  3     ! All requests that no longer have operator coverage will be canceled.
;  824          0818  3     !
;  825          0819  3     IF .LOST_COVERAGE
;  826          0820  3     THEN
;  827          0821  3         CHECK_OPER_COVERAGE (.OCD);
;  828          0822  2     END;
;  829          0823  2
;  830          0824  2 RETURN;                                          ! Return the routine value
;  831          0825  2
;  832          0826  1 END;                                             ! End of OPERUTIL_CLM_IMP_DISABLE


                                                                          .PSECT  $PLIT$,NOWRT,NOEXE,2

20 45 4C 42 41 4E 45 52 50 4F 5F 5F 4D 4C 43 00024 P.AAD:  .ASCII  \CLM__OPRENABLE mismatch\<0>                    ;
                  00 68 63 74 61 6D 73 69 6D 00033
```

```
                                010E0017  0003C P.AAC:  .LONG   17694743                                          ;
                                00000000' 00040         .ADDRESS P.AAD                                         ;

                                                        .EXTRN  CLUSMSG_CONV_CLM_RQCB
                                                        .EXTRN  ASCID_INVALIDRQCB

                                                        .PSECT  $CODE$,NOWRT,2

                                    000C 00000          .ENTRY  OPERUTIL_CLM_IMP_DISABLE, Save R2,R3           ; 0720
                          5E    FF70   CE   9E 00002    MOVAB   -144(SP), SP                                   ;
                          52      08   AC   D0 00007    MOVL    CLM, R2                                        ; 0781
                          02      02   A2   91 0000B    CMPB    2(R2), #2                                      ;
                                       06   13 0000F    BEQL    1$                                             ;
                                0000'  CF   9F 00011    PUSHAB  P.AAC                                          ; 0783
                                       10   11 00015    BRB     2$                                             ;
                                4004   8F   BB 00017 1$: PUSHR   #^M<R2,SP>                                   ; 0787
                0000G     CF           02   FB 0001B    CALLS   #2, CLUSMSG_CONV_CLM_RQCB                      ;
                          0D           50   E8 00020    BLBS    R0, 3$                                         ;
                                0000G  CF   9F 00023    PUSHAB  ASCID_INVALIDRQCB                              ; 0789
                                  04   AC   DD 00027 2$: PUSHL   BUFFER_DESC                                   ;
                0000G     CF           02   FB 0002A    CALLS   #2, DUMP_LOG_FILE                              ;
                                       04   0002F        RET                                                    ;
                                  04   AE   9F 00030 3$: PUSHAB  OPER_RQCB                                     ; 0795
                                  04   AE   DD 00033    PUSHL   RQCB                                           ;
                FD1D      CF           02   FB 00036    CALLS   #2, FIND_OPERATOR                              ;
                          52           50   D0 0003B    MOVL    R0, FOUND                                      ;
                                       6E   DD 0003E    PUSHL   RQCB                                           ; 0796
                0000G     CF           01   FB 00040    CALLS   #1, DEALLOCATE_RQCB                            ;
                          34           52   E9 00045    BLBC    FOUND, 4$                                      ; 0802
                          52      04   AE   D0 00048    MOVL    OPER_RQCB, R2                                  ; 0808
                                       52   DD 0004C    PUSHL   R2                                             ;
                          7E      5C   A2   7D 0004E    MOVQ    92(R2), -(SP)                                  ; 0806
                                       01   DD 00052    PUSHL   #1                                             ; 0805
                0000G     CF           04   FB 00054    CALLS   #4, UPD_OPER_CONTEXT                           ;
                          53           50   D0 00059    MOVL    R0, LOST_COVERAGE                              ;
                            04         62   0F 0005C    REMQUE  (R2), OPER_RQCB                                ; 0810
                          50      04   AE   D0 00060    MOVL    OPER_RQCB, R0                                  ; 0811
                          52      24   A0   D0 00064    MOVL    36(R0), OCD                                    ;
                                  46   A2   B7 00068    DECW    70(OCD)                                        ; 0812
                                       50   DD 0006B    PUSHL   R0                                             ; 0813
                0000G     CF           01   FB 0006D    CALLS   #1, DEALLOCATE_RQCB                            ;
                          07           53   E9 00072    BLBC    LOST_COVERAGE, 4$                              ; 0819
                                       52   DD 00075    PUSHL   OCD                                            ; 0821
                0000G     CF           01   FB 00077    CALLS   #1, CHECK_OPER_COVERAGE                        ;
                                       04   0007C 4$:   RET                                                    ; 0826

; Routine Size:  125 bytes,    Routine Base:  $CODE$ + 0309
```

```
;  834     0827  1 GLOBAL ROUTINE UPD_OPER_CONTEXT (DISABLE, MASK1, MASK2, RQCB) =
;  835     0828  1
;  836     0829  1 !++
;  837     0830  1 ! Functional description:
;  838     0831  1 !
;  839     0832  1 !       Update the OCD count vector for each bit present in the bit mask.
;  840     0833  1 !       The count will be decremented for a DISABLE, incremented for an ENABLE.
;  841     0834  1 !       Also update the OCD operator intrest mask, and the corresponding intrest
;  842     0835  1 !       mask in the operator RQCB.  This must be done in two loops, due to
;  843     0836  1 !       BLISS's inability to cope with a bitmask of more then 32 elements.
;  844     0837  1 !       Also note that the code could be more compact, but I traded that
;  845     0838  1 !       for readability.
;  846     0839  1 !
;  847     0840  1 ! Input:
;  848     0841  1 !
;  849     0842  1 !       DISABLE : A boolean value that declares whether this is an ENABLE or DISABLE.
;  850     0843  1 !       MASK1   : The first 32 bits of an operator attention mask.
;  851     0844  1 !       MASK2   : The second 32 bits of an operator attention mask.
;  852     0845  1 !       RQCB    : Address of an operator RQCB.
;  853     0846  1 !
;  854     0847  1 ! Implicit Input:
;  855     0848  1 !
;  856     0849  1 !       None.
;  857     0850  1 !
;  858     0851  1 ! Output:
;  859     0852  1 !
;  860     0853  1 !       None.
;  861     0854  1 !
;  862     0855  1 ! Implict output:
;  863     0856  1 !
;  864     0857  1 !       The operator context contained in the RQCB
;  865     0858  1 !       and the appropriate OCD is updated.
;  866     0859  1 !
;  867     0860  1 ! Side effects:
;  868     0861  1 !
;  869     0862  1 !       None.
;  870     0863  1 !
;  871     0864  1 ! Routine value:
;  872     0865  1 !
;  873     0866  1 !       TRUE    : If an element of the countvector went to 0
;  874     0867  1 !       FALSE   : If no element of the countvector went to 0
;  875     0868  1 !--
;  876     0869  1
;  877     0870  2 BEGIN                                           ! Start of UPD_OPER_CONTEXT
;  878     0871  2
;  879     0872  2 MAP
;  880     0873  2         RQCB            : $ref_bblock;          ! Operator RQCB
;  881     0874  2
;  882     0875  2 LOCAL
;  883     0876  2         OCD             : $ref_bblock,          ! OCD data structure
;  884     0877  2         K               : LONG,                 ! Index into enablecount vector
;  885     0878  2         TRANSITION      : LONG,                 ! Boolean
;  886     0879  2         ENABLE_MASK     : BITVECTOR [32],       ! ENABLE/DISABLE control bits
;  887     0880  2         CHANGE_BITS1    : LONG,                 ! ditto
;  888     0881  2         CHANGE_BITS2    : LONG;                 ! ibid
;  889     0882  2
;  890     0883  2 TRANSITION = FALSE;
```

```
;  891          0884  2 IF .DISABLE
;  892          0885  2 THEN
;  893          0886  2     BEGIN
;  894          0887  2     !
;  895          0888  2     ! This is a DISABLE request.  Determine the bits to clear.
;  896          0889  2     !
;  897          0890  2     CHANGE_BITS1 = .RQCB [RQCB_L_ATTNMASK1] AND .MASK1;
;  898          0891  2     CHANGE_BITS2 = .RQCB [RQCB_L_ATTNMASK2] AND .MASK2;
;  899          0892  2     END
;  900          0893  2 ELSE
;  901          0894  2     BEGIN
;  902          0895  2     !
;  903          0896  2     ! This is an ENABLE request.  Determine the bits to set.
;  904          0897  2     !
;  905          0898  2     CHANGE_BITS1 = (NOT .RQCB [RQCB_L_ATTNMASK1]) AND .MASK1;
;  906          0899  2     CHANGE_BITS2 = (NOT .RQCB [RQCB_L_ATTNMASK2]) AND .MASK2;
;  907          0900  2     END;
;  908          0901  2
;  909          0902  2 !
;  910          0903  2 ! Get the OCD address and do the update.
;  911          0904  2 !
;  912          0905  2 OCD = .RQCB [RQCB_L_OCD];                    ! Get OCD address
;  913          0906  2 ENABLE_MASK = .CHANGE_BITS1;                 ! Get first 32 bits
;  914          0907  2 INCR J FROM 0 TO 31 DO
;  915          0908  2     IF .ENABLE_MASK [.J]
;  916          0909  2     THEN
;  917          0910  2         IF .DISABLE
;  918          0911  2         THEN
;  919          0912  3             BEGIN
;  920          0913  3             RQCB [RQCB_L_ATTNMASK1] = .RQCB [RQCB_L_ATTNMASK1] AND (NOT (1^.J));
;  921          0914  3             OCD [OCD_W_ENABLECOUNT (.J)] = .OCD [OCD_W_ENABLECOUNT (.J)] - 1;
;  922          0915  4             IF (.OCD [OCD_W_ENABLECOUNT (.J)] EQL 0)
;  923          0916  3             THEN
;  924          0917  4                 BEGIN
;  925          0918  4                 TRANSITION = TRUE;
;  926          0919  4                 OCD [OCD_L_ATTNMASK1] = .OCD [OCD_L_ATTNMASK1] AND (NOT (1^.J));
;  927          0920  3                 END;
;  928          0921  3             END
;  929          0922  2         ELSE
;  930          0923  3             BEGIN
;  931          0924  3             RQCB [RQCB_L_ATTNMASK1] = .RQCB [RQCB_L_ATTNMASK1] OR (1^.J);
;  932          0925  3             OCD [OCD_W_ENABLECOUNT (.J)] = .OCD [OCD_W_ENABLECOUNT (.J)] + 1;
;  933          0926  3             OCD [OCD_L_ATTNMASK1] = .OCD [OCD_L_ATTNMASK1] OR (1^.J);
;  934          0927  2             END;
;  935          0928  2
;  936          0929  2 ENABLE_MASK = .CHANGE_BITS2;                 ! Get second 32 bits
;  937          0930  2 INCR J FROM 0 TO 31 DO
;  938          0931  2     IF .ENABLE_MASK [.J]
;  939          0932  2     THEN
;  940          0933  3         BEGIN
;  941          0934  3         K = .J + 32;
;  942          0935  3         IF .DISABLE
;  943          0936  3         THEN
;  944          0937  4             BEGIN
;  945          0938  4             RQCB [RQCB_L_ATTNMASK2] = .RQCB [RQCB_L_ATTNMASK2] AND (NOT (1^.J));
;  946          0939  4             OCD [OCD_W_ENABLECOUNT (.K)] = .OCD [OCD_W_ENABLECOUNT (.K)] - 1;
;  947          0940  5             IF (.OCD [OCD_W_ENABLECOUNT (.K)] EQL 0)
```

```
;  948          0941  4              THEN
;  949          0942  5                  BEGIN
;  950          0943  5                  TRANSITION = TRUE;
;  951          0944  5                  OCD [OCD_L_ATTNMASK2] = .OCD [OCD_L_ATTNMASK2] AND (NOT (1^.J));
;  952          0945  4                  END;
;  953          0946  4              END
;  954          0947  3          ELSE
;  955          0948  4              BEGIN
;  956          0949  4              RQCB [RQCB_L_ATTNMASK2] = .RQCB [RQCB_L_ATTNMASK2] OR (1^.J);
;  957          0950  4              OCD [OCD_W_ENABLECOUNT (.K)] = .OCD [OCD_W_ENABLECOUNT (.K)] + 1;
;  958          0951  4              OCD [OCD_L_ATTNMASK2] = .OCD [OCD_L_ATTNMASK2] OR (1^.J);
;  959          0952  3              END;
;  960          0953  2          END;
;  961          0954  2
;  962          0955  2 RETURN (.TRANSITION);
;  963          0956  2
;  964          0957  1 END;                                    ! End of UPD_OPER_CONTEXT


                                  01FC 00000          .ENTRY  UPD_OPER_CONTEXT, Save R2,R3,R4,R5,R6,R7,R8 ; 0827
                               58  D4 00002          CLRL    TRANSITION                                  ; 0883
                          50   10  AC  D0 00004          MOVL    RQCB, R0                                    ; 0890
                          53   5C  A0  9E 00008          MOVAB   92(R0), R3                                  ;
                          55   60  A0  9E 0000C          MOVAB   96(R0), R5                                  ; 0891
                          12   04  AC  E9 00010          BLBC    DISABLE, 1$                                 ; 0884
                          51   08  AC  D2 00014          MCOML   MASK1, CHANGE_BITS1                         ; 0890
                     51   63       51  CB 00018          BICL3   CHANGE_BITS1, (R3), CHANGE_BITS1            ;
                          56   0C  AC  D2 0001C          MCOML   MASK2, CHANGE_BITS2                         ; 0891
                     56   65       56  CB 00020          BICL3   CHANGE_BITS2, (R5), CHANGE_BITS2            ;
                                   0A  11 00024          BRB     2$                                          ; 0884
                     51        08 AC  63  CB 00026 1$:     BICL3   (R3), MASK1, CHANGE_BITS1                   ; 0898
                     56        0C AC  65  CB 0002B          BICL3   (R5), MASK2, CHANGE_BITS2                   ; 0899
                          52   24  A0  D0 00030 2$:     MOVL    36(R0), OCD                                 ; 0905
                          57       51  D0 00034          MOVL    CHANGE_BITS1, ENABLE_MASK                   ; 0906
                                   51  D4 00037          CLRL    J                                           ; 0907
                     32   57       51  E1 00039 3$:     BBC     J, ENABLE_MASK, 5$                          ; 0908
                          50   58 A241  3E 0003D          MOVAW   B8(OCD)[J], R0                              ; 0914
                          18   04  AC  E9 00042          BLBC    DISABLE, 4$                                 ;
                     54   01       51  78 00046          ASHL    J, #1, R4                                   ; 0913
                          63       54  CA 0004A          BICL2   R4, (R3)                                    ;
                                   60  B7 0004D          DECW    (R0)                                        ; 0914
                                   1E  12 0004F          BNEQ    5$                                          ; 0915
                          58       01  D0 00051          MOVL    #1, TRANSITION                              ; 0918
                     50   01       51  78 00054          ASHL    J, #1, R0                                   ; 0919
                               48  A2  50  CA 00058          BICL2   R0, 72(OCD)                                 ;
                                   11  11 0005C          BRB     5$                                          ; 0910
                     54   01       51  78 0005E 4$:     ASHL    J, #1, R4                                   ; 0924
                          63       54  C8 00062          BISL2   R4, (R3)                                    ;
                                   60  B6 00065          INCW    (R0)                                        ; 0925
                     50   01       51  78 00067          ASHL    J, #1, R0                                   ; 0926
                               48  A2  50  C8 0006B          BISL2   R0, 72(OCD)                                 ;
                     C6   51       1F  F3 0006F 5$:     AOBLEQ  #31, J, 3$                                  ; 0908
                          57       56  D0 00073          MOVL    CHANGE_BITS2, ENABLE_MASK                   ; 0929
                                   51  D4 00076          CLRL    J                                           ; 0930
```

```
                36          57          51  E1 00078 6$:     BBC     J, ENABLE_MASK, 8$                  ; 0931
                            50      20  A1  9E 0007C         MOVAB   32(R1), K                           ; 0934
                            53  58  A240  3E 00080         MOVAW   88(OCD)[K], R3                      ; 0939
                            18      04  AC  E9 00085         BLBC    DISABLE, 7$                         ;
                54          01          51  78 00089         ASHL    J, #1, R4                           ; 0938
                            65          54  CA 0008D         BICL2   R4, (R5)                            ;
                                        63  B7 00090         DECW    (R3)                                ; 0939
                                        1E  12 00092         BNEQ    8$                                  ; 0940
                            58          01  D0 00094         MOVL    #1, TRANSITION                      ; 0943
                53          01          51  78 00097         ASHL    J, #1, R3                           ; 0944
                        4C  A2          53  CA 0009B         BICL2   R3, 76(OCD)                         ;
                                        11  11 0009F         BRB     8$                                  ; 0935
                54          01          51  78 000A1 7$:     ASHL    J, #1, R4                           ; 0949
                            65          54  C8 000A5         BISL2   R4, (R5)                            ;
                                        63  B6 000A8         INCW    (R3)                                ; 0950
                53          01          51  78 000AA         ASHL    J, #1, R3                           ; 0951
                        4C  A2          53  C8 000AE         BISL2   R3, 76(OCD)                         ;
                C2          51          1F  F3 000B2 8$:     AOBLEQ  #31, J, 6$                          ; 0931
                            50          58  D0 000B6         MOVL    TRANSITION, R0                      ; 0955
                                        04  000B9         RET                                         ; 0957

; Routine Size:  186 bytes,    Routine Base:  $CODE$ + 0386
```

```
;  966     0958  1 GLOBAL ROUTINE VALID_OPERATOR (BUFFER_DESC, RQCB) =
;  967     0959  1
;  968     0960  1 !++
;  969     0961  1 ! Functional description:
;  970     0962  1 !
;  971     0963  1 !       This routine will make sure that the device
;  972     0964  1 !       specified in the user's request is capable
;  973     0965  1 !       of being an operator device.  A side effect
;  974     0966  1 !       of this routine is to create an operator device
;  975     0967  1 !       name descriptor within the RQCB.  Note that
;  976     0968  1 !       the operator device name is formatted in such
;  977     0969  1 !       a way as to make for easy string compares in
;  978     0970  1 !       the future.
;  979     0971  1 !
;  980     0972  1 ! Input:
;  981     0973  1 !
;  982     0974  1 !       BUFFER_DESC     : Address of string descriptor that points
;  983     0975  1 !                         to the user's request message.
;  984     0976  1 !       RQCB            : Address of an RQCB data structure.
;  985     0977  1 !
;  986     0978  1 ! Implicit Input:
;  987     0979  1 !
;  988     0980  1 !       None.
;  989     0981  1 !
;  990     0982  1 ! Output:
;  991     0983  1 !
;  992     0984  1 !       None.
;  993     0985  1 !
;  994     0986  1 ! Implict output:
;  995     0987  1 !
;  996     0988  1 !       None.
;  997     0989  1 !
;  998     0990  1 ! Side effects:
;  999     0991  1 !
; 1000     0992  1 !       A string descriptor of the validated operator device name
; 1001     0993  1 !       is created within the RQCB.
; 1002     0994  1 !
; 1003     0995  1 ! Routine value:
; 1004     0996  1 !
; 1005     0997  1 !       TRUE    : If the device is a valid operator device
; 1006     0998  1 !       FALSE   : If the device is not a valid operator device.
; 1007     0999  1 !--
; 1008     1000  1
; 1009     1001  2 BEGIN                                           ! Start of VALID_OPERATOR
; 1010     1002  2
; 1011     1003  2 MAP
; 1012     1004  2         BUFFER_DESC     : $ref_bblock,          ! User's request descriptor
; 1013     1005  2         RQCB            : $ref_bblock;          ! RQCB data structure
; 1014     1006  2
; 1015     1007  2 EXTERNAL
; 1016     1008  2         DEVICE_FAO      : $bblock;              ! FAO control string descriptor
; 1017     1009  2
; 1018     1010  2 EXTERNAL ROUTINE
; 1019     1011  2         SHARE_FULL_DEVNAME;                     ! Expand device name
; 1020     1012  2
; 1021     1013  2 LOCAL
; 1022     1014  2         ARG_LIST        : VECTOR [4],           ! Arguement list structure
```

```
;1023     1015  2          ARB             : $bblock [ARB$K_LENGTH],! Access rights block
;1024     1016  2          MSG             : $ref_bblock,          ! Pointer to user request
;1025     1017  2          DEV_CHAR        : $bblock [DIB$K_LENGTH],! Dev. char. buffer
;1026     1018  2          CHAR_DESC       : $desc_block,          ! Dev. char. buffer descriptor
;1027     1019  2          FULL_DESC       : $ref_bblock,          ! Descriptor for expanded name
;1028     1020  2          OPR_NAM_BUF     : $bblock [MAX_DEV_NAM],! Oper. device name buffer
;1029     1021  2          OPR_NAM_DESC    : $desc_block,          ! Oper. dev. name buffer descriptor
;1030     1022  2          STATUS          : LONG;
;1031     1023  2      !
;1032     1024  2      ! See if the requestor is issuing this request in another's behalf.
;1033     1025  2      ! If, and the requestor does not have the privilege to do so, then
;1034     1026  2      ! return FALSE.  Allow the request if the requestor has OPER pivilege,
;1035     1027  2      ! or the GROUP field of the UIC's are the same and the requestor has
;1036     1028  2      ! GROUP privilege.
;1037     1029  2      !
;1038     1030  2      IF .RQCB [RQCB_L_SENDERUIC] NEQ .RQCB [RQCB_L_UIC]
;1039     1031  2      THEN
;1040     1032  3          IF  (NOT .$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER])
;1041     1033  2          THEN
;1042     1034  3              IF NOT ((.$bblock [RQCB [RQCB_L_SENDERUIC], 2,0,16,0] EQL .$bblock [RQCB [RQCB_L_UIC], 2,0,16,0]) AN
;1043     1035  3                      (.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_GROUP]))
;1044     1036  2              THEN
;1045     1037  2                  RETURN (FALSE);
;1046     1038  2      !
;1047     1039  2      ! Create a descriptor for the operator device name.
;1048     1040  2      !
;1049     1041  2      MSG = .BUFFER_DESC [DSC$A_POINTER] + OPC$K_COMHDRSIZ;
;1050     1042  2      OPR_NAM_DESC [0,0,32,0] = .$bblock [MSG [OPC$T_OPRENABLE_OPR], 0,0,8,0];
;1051     1043  2      OPR_NAM_DESC [DSC$A_POINTER] = MSG [OPC$T_OPRENABLE_OPR] + 1;
;1052     1044  2      !
;1053     1045  2      ! Create a buffer descriptor and get the device
;1054     1046  2      ! characteristics of the operator device.
;1055     1047  2      !
;1056     1048  2      CHAR_DESC [0,0,32,0] = DIB$K_LENGTH;
;1057     1049  2      CHAR_DESC [DSC$A_POINTER] = DEV_CHAR;
;1058     1050  3      IF NOT (STATUS = $GETDEV (DEVNAM=OPR_NAM_DESC, PRIBUF=CHAR_DESC))
;1059     1051  2      THEN
;1060     1052  2          RETURN (.STATUS);                           ! There is no such device
;1061     1053  2      !
;1062     1054  2      ! Check the device type.  The device must be a
;1063     1055  2      ! terminal, remote terminal, or mailbox.
;1064     1056  2      !
;1065     1057  2      IF (NOT .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_TRM]) AND
;1066     1058  3         (NOT .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_MBX])
;1067     1059  2      THEN
;1068     1060  2          RETURN (FALSE);
;1069     1061  2      !
;1070     1062  2      ! If the device is a mailbox, then indicate such
;1071     1063  2      ! and save the device buffer size.  The requestor
;1072     1064  2      ! must have read and write access to the mailbox.
;1073     1065  2      !
;1074     1066  2      IF .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_MBX]
;1075     1067  2      THEN
;1076     1068  3          BEGIN
;1077     1069  3          RETURN (FALSE);
;1078     1070  3      !
;1079     1071  3      ! The mailbox as operator implementation is not complete.  Tie off this code by
```

```
;  1080          1072  3 ! commenting it out.
;  1081          1073  3 !
;  1082          1074  3 !       RQCB [OPRSTS_V_MBX] = TRUE;                             ! Mark OPER as MBX
;  1083          1075  3 !       RQCB [RQCB_W_MBXSIZE] = .DEV_CHAR [DIB$W_DEVBUFSIZ];! Save MBX size
;  1084          1076  3 !       !
;  1085          1077  3 !       ! The following code is a workaround until a GETACCess
;  1086          1078  3 !       ! system service can be written.  Check for R/W access.
;  1087          1079  3 !       !
;  1088          1080  3 !       CH$FILL (0, ARB$K_LENGTH, ARB);                         ! Fill with blanks
;  1089          1081  3 !       (ARB [ARB$Q_PRIV])    = .RQCB [RQCB_L_ATTNMASK1];       ! Build a dummy ARB
;  1090          1082  3 !       (ARB [ARB$Q_PRIV]+4) = .RQCB [RQCB_L_ATTNMASK2];        !
;  1091          1083  3 !       ARB [ARB$L_UIC]      = .RQCB [RQCB_L_UIC];              !
;  1092          1084  3 !       ARG_LIST [0] = 3;                                       ! Build an argument list
;  1093          1085  3 !       ARG_LIST [1] = ARB;                                     !  Address of ARB
;  1094          1086  3 !       ARG_LIST [2] = .DEV_CHAR [DIB$W_VPROT];                 ! Volume protection mask
;  1095          1087  3 !       ARG_LIST [2] = .DEV_CHAR [DIB$L_OWNUIC];                ! Volume owner
;  1096          1088  3 !       IF NOT (STATUS = $CMKRNL (ROUTIN=EXE$CHKRDACCES, ARGLST=ARG_LIST))
;  1097          1089  3 !       OR NOT (STATUS = $CMKRNL (ROUTIN=EXE$CHKWRTACCES, ARGLST=ARG_LIST))
;  1098          1090  3 !       THEN
;  1099          1091  3 !           RETURN (.STATUS);                                   ! No R/W access
;  1100          1092  2 !       END;
;  1101          1093  2 !
;  1102          1094  2 !
;  1103          1095  2 ! If the device is terminal, mark it as such.  If it is
;  1104          1096  2 ! a remote terminal or a dial-in terminal, then mark it
;  1105          1097  2 ! as a remote terminal.
;  1106          1098  2 !
;  1107          1099  2 ! NOTE: THE METHOD OF DETERMINING IF A TERMINAL IS
;  1108          1100  2 !       A REMOTE TERMINAL MAY CHANGE OVER TIME.
;  1109          1101  2 !
;  1110          1102  2 !
;  1111          1103  2 IF .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_TRM]
;  1112          1104  2 THEN
;  1113          1105  2     IF .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_MNT]
;  1114          1106  2     OR .$bblock [DEV_CHAR [DIB$L_DEVDEPEND], TT$V_MODEM]
;  1115          1107  2     THEN
;  1116          1108  2         RQCB [OPRSTS_V_REMTRM] = TRUE
;  1117          1109  2     ELSE
;  1118          1110  2         RQCB [OPRSTS_V_TRM] = TRUE;
;  1119          1111  2 !
;  1120          1112  2 ! Format the operator device name from the info
;  1121          1113  2 ! in the device characteristics buffer.  All operator
;  1122          1114  2 ! devices known to OPCOM have their operator device
;  1123          1115  2 ! names formatted here, so that they are in a consistant
;  1124          1116  2 ! format.
;  1125          1117  2 !
;  1126          1118  2 OPR_NAM_DESC [0,0,32,0] = MAX_DEV_NAM;           ! Create an output string descriptor
;  1127          1119  2 OPR_NAM_DESC [DSC$A_POINTER] = OPR_NAM_BUF;      !
;  1128        P 1120  3 IF NOT (STATUS = $FAO   (DEVICE_FAO,             ! Format the operator device name
;  1129        P 1121  3                          OPR_NAM_DESC,           !
;  1130        P 1122  3                          OPR_NAM_DESC,           !
;  1131        P 1123  3                          DEV_CHAR + .DEV_CHAR [DIB$W_DEVNAMOFF],
;  1132        P 1124  3                          .DEV_CHAR [DIB$Q_UNIT]
;  1133          1125  3                         ))
;  1134          1126  2 THEN
;  1135          1127  2     RETURN (.STATUS);
;  1136          1128  2 !
```

```
;  1137     1129  2 ! Expand the name to the full name
;  1138     1130  2 !
;  1139     1131  2 FULL_DESC = SHARE_FULL_DEVNAME (OPR_NAM_DESC, DVI$_FULLDEVNAM);
;  1140     1132  2 RQCB [RQCB_L_OPER_LEN] = .FULL_DESC [DSC$W_LENGTH];
;  1141     1133  2 !
;  1142     1134  2 ! Create a string descriptor for the formatted
;  1143     1135  2 ! operator device name, within the RQCB.
;  1144     1136  2 !
;  1145     1137  2 IF NOT (STATUS = OPC$GET_VM (RQCB [RQCB_L_OPER_LEN], RQCB [RQCB_L_OPER_PTR]))
;  1146     1138  2 THEN
;  1147     1139  2     RETURN (.STATUS);
;  1148     1140  2 !
;  1149     1141  2 ! Copy the operator device name to the new buffer.
;  1150     1142  2 !
;  1151     1143  2 CH$MOVE (.RQCB [RQCB_L_OPER_LEN],
;  1152     1144  2           .FULL_DESC [DSC$A_POINTER],
;  1153     1145  2           .RQCB [RQCB_L_OPER_PTR]
;  1154     1146  2          );
;  1155     1147  2 RETURN (TRUE);
;  1156     1148  2
;  1157     1149  1 END;                                   ! End of VALID_OPERATOR


                                                        .EXTRN  DEVICE_FAO, SHARE_FULL_DEVNAME
                                                        .EXTRN  SYS$FAO, OPC$GET_VM

                                        003C 00000      .ENTRY  VALID_OPERATOR, Save R2,R3,R4,R5        ; 0958
                                5E      FEB4 CE  9E 00002       MOVAB   -332(SP), SP                    ;
                                52        08 AC  D0 00007       MOVL    RQCB, R2                        ; 1030
                                51        38 A2  9E 0000B       MOVAB   56(R2), R1                      ;
                                50        68 A2  9E 0000F       MOVAB   104(R2), R0                     ;
                                60           61  D1 00013       CMPL    (R1), (R0)                      ;
                                             13  13 00016       BEQL    3$                              ;
                             0E           32 A2  02 E0 00018    BBS     #2, 50(R2), 3$                  ; 1032
                                          02 A0  02 A1 B1 0001D CMPW    2(R1), 2(R0)                    ; 1034
                                             03  13 00022       BEQL    2$                              ;
                                        00BB     31 00024 1$:    BRW     9$                              ;
                                           F9 31 A2  E9 00027 2$:    BLBC    49(R2), 1$                      ; 1035
                                           50 04 AC  D0 0002B 3$:    MOVL    BUFFER_DESC, R0                 ; 1041
                                50         04 A0  26 C1 0002F       ADDL3   #38, 4(R0), MSG                 ;
                                           6E 1A A0  9A 00034       MOVZBL  26(MSG), OPR_NAM_DESC           ; 1042
                                           04 AE 1B A0  9E 00038    MOVAB   27(R0), OPR_NAM_DESC+4          ; 1043
                                           48 AE 74 8F  9A 0003D    MOVZBL  #116, CHAR_DESC                 ; 1048
                                           4C AE 50 AE  9E 00042    MOVAB   DEV_CHAR, CHAR_DESC+4           ; 1049
                                              7E  7C 00047          CLRQ    -(SP)                           ; 1050
                                              50 AE  9F 00049       PUSHAB  CHAR_DESC                       ;
                                              7E  D4 0004C          CLRL    -(SP)                           ;
                                              10 AE  9F 0004E       PUSHAB  OPR_NAM_DESC                    ;
                                00000000G     00  FB 00051       CALLS   #5, SYS$GETDEV                  ;
                                              54  D0 00058       MOVL    R0, STATUS                      ;
                                              74  E9 0005B       BLBC    STATUS, 7$                      ;
                             7F           50 AE  E1 0005E       BBC     #2, DEV_CHAR, 9$                ; 1057
                             7A           52 AE  E0 00063       BBS     #4, DEV_CHAR+2, 9$              ; 1066
                             14           50 AE  E1 00068       BBC     #2, DEV_CHAR, 6$                ; 1103
                             05           52 AE  E0 0006D       BBS     #3, DEV_CHAR+2, 4$              ; 1105
                             06           5A AE  E1 00072       BBC     #5, DEV_CHAR+10, 5$             ; 1106
```

```
                    78   A2              02  BB 00077 4$:    BISB2   #2, 120(R2)                          ; 1108
                                         04  11 0007B        BRB     6$                                  ;
                    78   A2              01  BB 0007D 5$:    BISB2   #1, 120(R2)                         ; 1110
                    6E                40  8F  9A 00081 6$:    MOVZBL  #64, OPR_NAM_DESC                   ; 1118
                    04   AE          08  AE  9E 00085        MOVAB   OPR_NAM_BUF, OPR_NAM_DESC+4         ; 1119
                    7E               5C  AE  3C 0008A        MOVZWL  DEV_CHAR+12, -(SP)                  ; 1125
                    50               62  AE  3C 0008E        MOVZWL  DEV_CHAR+14, R0                     ;
                                     54 AE40 9F 00092        PUSHAB  DEV_CHAR[R0]                        ;
                                     08  AE  9F 00096        PUSHAB  OPR_NAM_DESC                        ;
                                     0C  AE  9F 00099        PUSHAB  OPR_NAM_DESC                        ;
                                   0000G CF  9F 0009C        PUSHAB  DEVICE_FAO                          ;
                00000000G  00              05  FB 000A0        CALLS   #5, SYS$FAO                         ;
                           54              50  D0 000A7        MOVL    R0, STATUS                          ;
                           25              54  E9 000AA        BLBC    STATUS, 7$                          ;
                           7E           E8  8F  9A 000AD        MOVZBL  #232, -(SP)                         ; 1131
                                        04  AE  9F 000B1        PUSHAB  OPR_NAM_DESC                        ;
                        0000G  CF              02  FB 000B4        CALLS   #2, SHARE_FULL_DEVNAME              ;
                           53              50  D0 000B9        MOVL    R0, FULL_DESC                       ;
                           7C   A2         63  3C 000BC        MOVZWL  (FULL_DESC), 124(R2)                ; 1132
                                      0080  C2  9F 000C0        PUSHAB  128(R2)                             ; 1137
                                        7C  A2  9F 000C4        PUSHAB  124(R2)                             ;
                        0000G  CF              02  FB 000C7        CALLS   #2, OPC$GET_VM                      ;
                           54              50  D0 000CC        MOVL    R0, STATUS                          ;
                           04              54  E8 000CF        BLBS    STATUS, 8$                          ;
                           50              54  D0 000D2 7$:    MOVL    STATUS, R0                          ; 1139
                                                04 000D5        RET                                         ;
          0080   D2        04   B3         7C  A2  28 000D6 8$:    MOVC3   124(R2), @4(FULL_DESC), @128(R2)    ; 1145
                           50              01  D0 000DE        MOVL    #1, R0                              ; 1147
                                                04 000E1        RET                                         ;
                                           50  D4 000E2 9$:    CLRL    R0                                  ; 1149
                                                04 000E4        RET                                         ;

; Routine Size:  229 bytes,    Routine Base:  $CODE$ + 0440

; 1158          1150  1
; 1159          1151  1 END                                             ! End of OPERUTIL
; 1160          1152  0 ELUDOM
```

```
;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $CODE$                         1317  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                           68  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                               Library Statistics
;
;                                -------- Symbols --------      Pages       Processing
```

```
;       File                                    Total    Loaded   Percent      Mapped     Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1                18619        32         0        1000     00:01.8
;  _$255$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1          633        52         8          43     00:00.9



;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:OPERUTIL/OBJ=OBJ$:OPERUTIL MSRC$:OPERUTIL/UPDATE=(ENH$:OPERUTIL)

; Size:         1317 code + 68 data bytes
; Run Time:       00:30.6
; Elapsed Time:   01:39.0
; Lines/CPU Min:   2259
; Lexemes/CPU-Min: 19902
; Memory Used:  157 pages
; Compilation Complete
```

LOGFILE
LIS

OPCOMDEF
LIS

OPCOMDATA
LIS

OPCOMINI
LIS

OPCOMUTIL
LIS

OPCOMLIB
LIS

OPCDEFTMP
LIS

OPCOMMAIN
LIS

OPCOMOLD
LIS

OPCOMRPLY
LIS

OPCCRASH
LIS

OPERUTIL
LIS

OPCOMRQST
LIS

REPLYBRD
LIS

SECURITY
LIS

OPRENABLE
LIS

REPLYMAIN
LIS

RQSTMAIN
LIS